Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA

GRADO EN INGENIERÍA INFORMÁTICA
MENCIÓN EN COMPUTACIÓN

Caso de estudio de aplicación de algoritmos genéticos para la optimización de rutas marítimas

---

Alumno:
Miguel Chaveinte García

Tutor:
Dña. Margarita Gonzalo Tasis

No confundas movimiento con progreso.
Peter Drucker

Ningún mar en calma hizo experto al marinero.
Cícero

El que controla el mar, lo controla todo.
Themistocles

Agradecimientos

A mis padres, mi hermana y familia por ser mi luz, mi guía y esa pata que me sostiene siempre. Y aunque no os lo diga todos los días: GRACIAS. Os quiero, y si una cosa tengo clara en estas páginas, es que me van a faltar palabras si hablamos de agradecéroslo.

A mis amigos, los que ya han estado en otras tantas, y para aquellos que han llegado nuevos en estos cuatro años. Gracias por formar parte de mi y de mi vida. Tengo un pedacito de vosotros en el corazón por siempre y este trabajo tiene una parte de vosotros. Orgulloso y muy feliz de teneros a mi lado.

A Margarita, mi tutora, por haber confiado en esta locura y haberme transmitido la paz que ni yo mismo sentía muchos días.

A mi mismo, por siempre intentar ser una mejor versión cada día, aunque a veces flaqueen las fuerzas. Aunque a veces la vida se ponga en contra. Aunque la cabeza falle. Siempre tendré claro que he sido lo que he querido ser.

Quizá el sentido de la vida sea construir, para destruir y comenzar de nuevo. No sé si este TFG es el principio de algo o el final, pero sí es todo lo que tengo, lo que soy y lo que quiero ser.

Resumen

La meteorología es un factor determinante para la rentabilidad, la seguridad y la sostenibilidad medioambiental de las rutas que recorren los buques de navegación marítima. Las condiciones meteorológicas afectan significativamente al rumbo de los buques, con implicaciones tanto para la seguridad de la tripulación como para el consumo de combustible y las emisiones contaminantes. Por ello, es necesario contar con un sistema eficaz de apoyo a la toma de decisiones que permita planificar la ruta y la velocidad del buque en función de las previsiones meteorológicas.

En este Trabajo de Fin de Grado se presenta la implementación de un modelo basado en algoritmos genéticos para minimizar el consumo de combustible y la diferencia entre la hora de llegada y la prevista de un buque , considerando las dos influencias meteorológicas más relevantes para la navegación: el viento y las olas. Nuestra propuesta ayuda a los planificadores de rutas a encontrar rutas de coste mínimo que tengan en cuenta la meteorología, eviten las zonas especificadas y cumplan las restricciones de tiempo de llegada.

Para ello se plantea un modelo de planificación de rutas multicriterio, cuyo enfoque sea la optimización de rutas marítimas, mediante el uso de los datos marítimos proporcionados por el Servicio de Vigilancia del Medio Marino de Copernicus (CMEMS) y AIS (Sistema de Identificación Automática), la aplicación de modelos de machine learning para predecir la velocidad del buque y la aplicación del algoritmo genético NSGA-II para obtener un conjunto de soluciones óptimas.

Abstract

Meteorology is a determining factor for the profitability, safety, and environmental sustainability of routes taken by maritime vessels. Weather conditions significantly impact the course of ships, with implications for both crew safety and fuel consumption as well as pollutant emissions. Therefore, it is necessary to have an effective decision support system that enables the planning of routes and vessel speeds based on weather forecasts.

This Final Degree Project presents the implementation of a model based on genetic algorithms to minimize fuel consumption and the difference between the actual arrival time and the estimated time of arrival of a ship, considering the two most relevant meteorological influences for navigation: wind and waves. Our proposal assists route planners in finding minimum-cost routes that take into account meteorology, avoid specified areas, and meet arrival time constraints.

To achieve this, a multi-criteria route planning model is proposed, focusing on the optimization of maritime routes. This is accomplished by utilizing maritime data provided by the Copernicus Marine Environment Monitoring Service (CMEMS) and the Automatic Identification System (AIS), applying machine learning models to predict vessel speed, and employing the NSGA-II genetic algorithm to obtain a set of optimal solutions.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Contexto

El transporte marítimo es un sector clave para la economía mundial, ya que representa alrededor del 80 % del comercio internacional y genera más de 13 millones de empleos directos e indirectos [15]. Sin embargo, también se enfrenta a grandes desafíos en términos de eficiencia, costes, seguridad y sostenibilidad. Los buques consumen grandes cantidades de combustible y emiten gases de efecto invernadero que contribuyen al cambio climático. Según la Organización Marítima Internacional (OMI)[16], el transporte marítimo es responsable del 2,5 % de las emisiones globales de CO_2 y tiene el potencial de reducirlas en un 50 % para 2050. Además, están expuestos a condiciones meteorológicas adversas que pueden afectar a su navegación y causar daños al buque, la tripulación o la carga. Según un estudio de Allianz Global Corporate & Specialty (AGCS) [17], casi la mitad de los accidentes marítimos con pérdidas humanas o de cargamento se deben al mal tiempo.

Tradicionalmente, la industria naviera ha dependido en gran medida de sus diversos mini-ecosistemas, que consisten en puertos, autoridades, barcos, operadores navieros, armadores, propietarios de carga y muchos más. Sin embargo, hace más de una década, los investigadores señalaron que la tradicionalmente conservadora industria naviera "está experimentando un cambio, donde se cree que las demandas de aumento de la eficiencia, la seguridad y la protección del medio ambiente solo se pueden lograr mediante más innovación" (Perunovic y Vidic, 2011)[18]. Más recientemente, el Boston Consulting Group identificó una serie de tecnologías emergentes como el análisis avanzado, la navegación autónoma, la robótica y la inteligencia artificial que están destinadas a cambiar la forma en que se realizan las funciones de planificación, operaciones comerciales y de apoyo dentro del transporte marítimo (Egloff, Sanders, Riedl, Mohottala y Georgaki, 2018)[19].

Con el progreso de la tecnología de navegación, los problemas de seguridad y ahorro de energía asociados a la navegación marítima se han convertido gradualmente en el centro de la atención humana. En este sentido, las rutas marítimas desempeñan un papel vital en el transporte marítimo y una ruta eficiente no sólo puede garantizar la seguridad de la navegación, sino que también aporta grandes beneficios económicos. Por lo tanto, el problema del cálculo de la ruta óptima es muy importante, especialmente para viajes largos. Por ejemplo, en los viajes transoceánicos, el tiempo de viaje es de varios días, y el consumo de combustible alcanza miles de toneladas. La planificación óptima de la ruta suele basarse en las previsiones meteorológicas, las condiciones del mar y las características del buque. Estos factores complican el proceso de determinación de la ruta óptima debido a diversas limitaciones como el litoral, las aguas poco profundas y las zonas prohibidas. La información sobre estas limitaciones pueden conocerse de antemano. Los condicionantes dinámicos, como el viento, el oleaje y otros factores, pueden predecirse mediante previsiones meteorológicas. Al mismo tiempo, los vientos favorables, las corrientes y otros

factores favorables pueden utilizarse para aumentar la velocidad de paso y reducir la duración del viaje.

Recientemente, la OMI y los gobiernos han prestado mucha atención a la contaminación atmosférica y al consumo de energía de los buques. En 2018, la OMI adoptó la "estrategia preliminar para la reducción de las emisiones de gases de efecto invernadero de los buques de la OMI", enviando así una fuerte señal a la comunidad internacional de que la industria naviera se está convirtiendo en una industria baja en carbono[20]. La reducción del consumo de combustible y de las emisiones de carbono mediante una planificación razonable de las rutas es una medida importante en respuesta a la estrategia de bajas emisiones de carbono. Para planificar una ruta eficiente, el capitán debe utilizar completamente los datos de previsión meteorológica en función de la misión del viaje y de las características del buque y evitar seleccionar rutas de alto riesgo frente a vientos y olas.

En las condiciones meteorológicas y marítimas imperantes, la "ruta óptima" puede definirse como la ruta con el tiempo medio de viaje mínimo, el consumo mínimo de combustible o cualquier combinación de ambos, garantizando al mismo tiempo la seguridad de la navegación. La determinación de la ruta óptima en función de las condiciones meteorológicas del buque requiere una combinación de las tres acciones siguientes (Sen and Padhy, 2015)[21]:

- Previsión de las condiciones del mar (es decir, previsión del estado del océano).
- Estimación del comportamiento de los buques en esas condiciones de oleaje oceánico.
- Desarrollo de un algoritmo de optimización de rutas o trayectorias adecuado y eficiente.

En la actualidad, el sector marítimo se enfrenta a una serie de desafíos que ponen en riesgo su sostenibilidad. En este contexto, la planificación de las rutas se ha convertido en una herramienta clave para garantizar la eficiencia y la rentabilidad de las operaciones marítimas; llevando a que la optimización de las rutas se haya convertido en una necesidad urgente para reducir el consumo de combustible y minimizar la huella de carbono, así como para garantizar la seguridad de los buques y la tripulación. Además, la gestión de riesgos asociados al medio ambiente, como la contaminación marina, es otra prioridad para la industria marítima. En este sentido, la tecnología y la IA se presentan como herramientas esenciales para la gestión logística del sector, permitiendo una planificación más precisa y eficiente de las rutas, lo que se traduce en una mejora en la eficiencia operativa y una reducción de los costos asociados.

1.2. Motivación

La planificación de las rutas siempre ha sido una cuestión crucial para la gestión logística del sector marítimo. Hoy en día, la optimización de las rutas está en juego para reducir el consumo de combustible, operar con seguridad los buques y abordar la gestión de riesgos teniendo en cuenta el medio ambiente.

La optimización de rutas marítimas, dentro de esta adquiere gran importancia ya que presenta un constante desafío para los operadores debido a la complejidad de los factores que influyen en la elección de la ruta más adecuada. Los servicios convencionales de enrutamiento meteorológico son útiles para evitar el "tiempo indeseado" y elegir rutas más seguras, pero suelen utilizar información genérica del buque, lo que limita su eficacia en términos de seguridad y eficiencia en el consumo de combustible y la reducción de emisiones.

Es por ello que la búsqueda de soluciones que permitan reducir el consumo de combustible, operar con seguridad los buques y minimizar los impactos ambientales se ha convertido en un objetivo clave en la planificación de rutas marítimas. Estas soluciones deben ser capaces de adaptarse a las condiciones cambiantes del entorno y de explorar el espacio de soluciones de forma inteligente y eficiente. En este

contexto, los algoritmos genéticos han demostrado ser una herramienta de gran potencial para la resolución de este problema, permitiendo encontrar soluciones eficientes y óptimas en términos de costos y tiempo.

En este sentido, el presente Trabajo de Fin de Grado tiene como objetivo profundizar en esta área de investigación y analizar las posibilidades de aplicación de los algoritmos genéticos en la optimización de rutas marítimas. Para ello, se propone la implementación de un modelo de simulación y la realización de diferentes experimentos y análisis comparativos, con el fin de evaluar la eficacia de los algoritmos genéticos en la mejora de la eficiencia en el consumo de combustible y la reducción de emisiones en el transporte marítimo.

La utilización de algoritmos genéticos en la optimización de rutas marítimas puede ser una solución innovadora y eficiente para el sector logístico marítimo, permitiendo mejorar la competitividad de las empresas y contribuir al cuidado del medio ambiente. Por lo tanto, el análisis de las posibilidades de aplicación de los algoritmos genéticos en la optimización de rutas marítimas se presenta como un tema de gran interés para el desarrollo de soluciones innovadoras en el sector marítimo, y es un área de investigación que requiere de un análisis profundo y riguroso.

1.3. Objetivos

El presente Trabajo Fin de Grado aborda la problemática de optimizar las rutas marítimas teniendo en cuenta los factores meteorológicos, operativos y económicos que influyen en la elección de la ruta más adecuada para cada operación. Para ello, se propone el uso de algoritmos genéticos como técnica de optimización inspirada en la evolución natural que permite encontrar soluciones eficientes y óptimas en términos de costos y tiempo. En esta línea de trabajo se identifican los siguientes objetivos a abordar:

1. Análisis de la problemática de la optimización de rutas marítimas
   - *a*) Identificación de los principales problemas asociados a la planificación de rutas marítimas, incluyendo la variabilidad del clima, las limitaciones de los buques y los desafíos operativos.
   - *b*) Revisión de las soluciones actuales para la optimización de rutas marítimas, incluyendo los enfoques basados en la teoría de grafos, la programación lineal y la metaheurística.
2. Procesamiento y Análisis de los datos de las rutas marítimas para crear un Modelo predictivo de características relevantes en la optimización de rutas.
   - *a*) Análisis de datos climatológicos y del AIS de los buques para la predicción de características relevantes en la optimización de rutas.
   - *b*) Definición del modelo de datos final que incluya las variables relevantes para el problema.
   - *c*) Extracción y preprocesamiento de los datos climatológicos y del AIS, de acuerdo al modelo de datos final.
   - *d*) Diseño y entrenamiento de modelos de aprendizaje automático para la predicción de características relevantes en la optimización de rutas.
   - *e*) Evaluación del desempeño de los modelos de predicción en términos de precisión y robustez.
3. Diseño y desarrollo de un modelo de simulación.
   - *a*) Definición de los parámetros y variables necesarios para la implementación del modelo de simulación.
   - *b*) Estudio de algoritmos genéticos que pueden utilizarse para la resolución del problema.

   c) Selección de los algoritmos genéticos más adecuados para la optimización de rutas marítimas, teniendo en cuenta las limitaciones y requerimientos específicos de la industria del transporte marítimo.
   d) Implementación del modelo de simulación y realización de pruebas para verificar su correcto funcionamiento.
   e) Refinamiento de los modelos de algoritmos genéticos realizando ajustes sobre sus parámetros para mejorar su desempeño final.

4. Realización de experimentos y análisis comparativos.

   a) Definición de los casos de prueba para la evaluación de las soluciones obtenidas a través del modelo de simulación.
   b) Aplicación de técnicas de evaluación y comparación para comprobar el desempeño real de los modelos de algoritmos genéticos refinados.
   c) Análisis de los resultados y conclusiones finales.

5. Diseño e implementación de una plataforma web interactiva para la optimización de rutas marítimas.

   a) Diseño y desarrollo de una interfaz de usuario intuitiva y fácil de usar para que el usuario pueda ingresar las características de su barco necesarias como variables para el modelo.
   b) Integración de los datos marítimos junto a los modelos de algoritmos genéticos para la optimización de rutas marítimas en la plataforma web.
   c) Desarrollo de una función de ajuste de parámetros para que el usuario pueda modificar los parámetros del modelo y obtener rutas optimizadas personalizadas.
   d) Integración de un sistema de representación gráfica en el mapa que muestre la ruta optimizada por el algoritmo genético y las condiciones climáticas actuales en tiempo real en el que se lleva a cabo la ruta.

La consecución de los objetivos planteados en este Trabajo de Fin de Grado permitirá la obtención de un modelo de algoritmo genético que tendrá la capacidad de optimizar las rutas marítimas teniendo en cuenta todos los factores relevantes, lo que implicará una mejora significativa en la eficiencia en el consumo de combustible y la reducción de emisiones en el transporte marítimo. En consecuencia, se busca aportar al avance del conocimiento en la materia y al desarrollo de soluciones que contribuyan al bienestar del planeta y a la sostenibilidad de las actividades económicas en el sector marítimo.

1.4. Soluciones: Software Similares

Para mejorar el rendimiento de los buques y reducir su impacto ambiental, es necesario optimizar las rutas marítimas que siguen. La optimización de rutas marítimas consiste en encontrar el itinerario más eficiente para un buque o una flota que debe realizar varias entregas o recogidas en diferentes puntos geográficos. La optimización de rutas marítimas implica tener en cuenta factores como la climatología, las corrientes marinas, los obstáculos, las restricciones de navegación, los costes de combustible y las ventanas horarias de los clientes.

Para optimizar las rutas marítimas de forma precisa y rápida, se necesita un software específico que pueda procesar toda esta información y ofrecer la mejor solución posible. En esta Sección, vamos a comparar cuatro software de optimización de rutas marítimas que se basan en diferentes tecnologías y ofrecen diferentes funcionalidades: Pythia, SPOS Seakeeping, Wayfinder y SINAY:

1.4.1. Pythia

Pythia [22] es una plataforma de DeepSea Technologies, una empresa de tecnología naval con sede en Londres y Atenas, que se utiliza para la optimización de rutas y el pronóstico del clima en la navegación marítima. Pythia utiliza modelos de inteligencia artificial (IA) para analizar el rendimiento exacto de cada buque bajo cualquier condición meteorológica y de incrustación (depósitos y sedimentos en el casco del buque). Pythia ofrece una optimización integral de la ruta, la velocidad y pautas de navegación para cada buque, con el objetivo de reducir el consumo de combustible y las emisiones de CO2.

Figura 1.1: Pythia platform user view

Pythia es utilizada por varias compañías navieras que buscan mejorar su eficiencia y sostenibilidad. Una de ellas es Seanergy Maritime Holdings Corp., una empresa líder en el sector de los capesize, que ha logrado un ahorro de combustible de hasta el 12 %, con un ahorro medio del 8 %, con Pythia en sus viajes durante los primeros cuatro meses de 2021 [23]. Algunas otras compañías utilizan Pythia como Danaos, Eurobulk LTD. o Wallenius Wilhelmsen (caso de estudio, con una mejora del 7 % en la eficiencia de media en sus 103 buques)[24]. Estas compañías eligen Pythia porque les permite obtener una comprensión precisa del rendimiento de sus buques y determinar las políticas óptimas de ruta, velocidad y navegación para cada viaje.

Pythia se basa en tecnologías avanzadas de IA, big data y computación en la nube. Pythia recoge datos en tiempo real de los sensores instalados en los buques, así como de fuentes externas como las previsiones meteorológicas o el estado de la mar (acuerdo con Spire Weather). Pythia procesa estos datos con algoritmos de aprendizaje profundo que crean modelos dinámicos y personalizados para cada buque. Pythia utiliza estos modelos para simular diferentes escenarios y recomendar la mejor opción para minimizar el consumo de combustible y las emisiones.

Son varios los puntos fuertes que la diferencian de otros software de rutas meteorológicas. Pythia es la primera plataforma que se adapta al rendimiento exacto de cada buque, teniendo en cuenta factores como el estado del casco, la carga y el lastre, en todas las condiciones utilizando modelos de rendimiento de IA basados en datos detallados en tiempo real. Esta información se utiliza para desarrollar un modelo de rendimiento dinámico y personalizado para cada buque.

La optimización ofrecida es integral que no se limita a la ruta, sino que también incluye la velocidad y el recorte (posición de flotación en dirección longitudinal). Pythia proporciona un seguimiento en tiempo real del buque y su entorno, así como un informe completo del rendimiento y los ahorros obtenidos. Lo que significa que, incluso teniendo en cuenta las incertidumbres meteorológicas, se puede predecir un

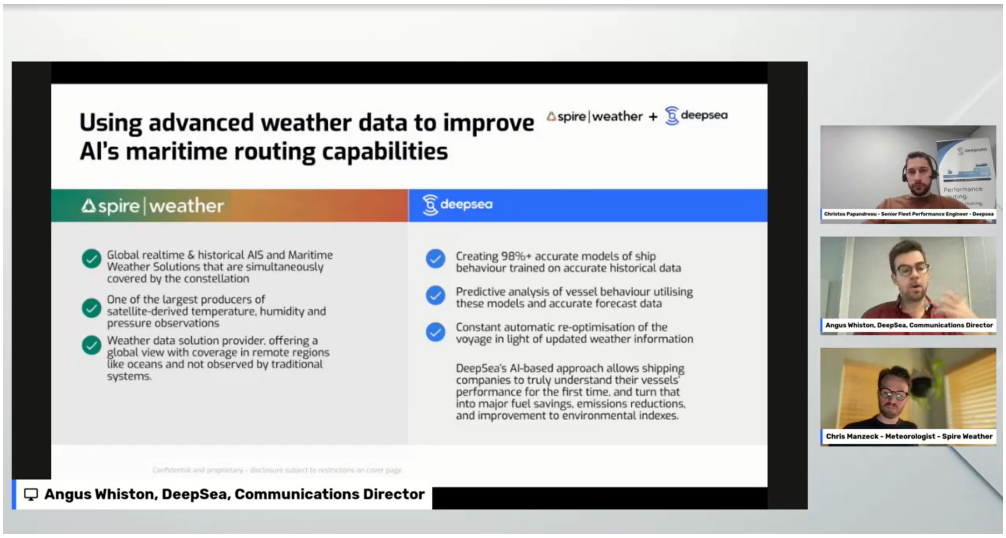

Figura 1.2: Puntos claves asociación Spire Weather x Deepsea

viaje -con una precisión del 99 %- antes de que comience. Además, la modelización se adapta en tiempo real a las condiciones cambiantes.[25]

Algunos de los puntos débiles Pythia depende de la calidad y la cantidad de los datos disponibles, lo que puede limitar su precisión y fiabilidad. La normalización de las flotas es clave para poder explotar al máximo el potencial de los datos.

Otro de los retos a la implementación de la IA es el cambio de distribución, que afecta a la evaluación del modelo y su robustez. El cambio de distribución tiene que ver con la falta de correspondencia entre los datos utilizados para entrenar un modelo y los datos que se encuentran en el mundo real.[26]

1.4.2. SPOS Seakeeping

SPOS Seakeeping [27] es un módulo adicional al sistema SPOS (Ship Performance Optimization System) desarrollado por DTN en colaboración con ABB. SPOS Seakeeping combina pronósticos meteorológicos precisos y detallados con modelado hidrodinámico y datos operativos de cada barco para aconsejar a los capitanes sobre la mejor ruta. SPOS Seakeeping tiene en cuenta el diseño del barco y su respuesta a las condiciones meteoceánicas al determinar la ruta óptima, lo que proporciona un asesoramiento más preciso y seguro adaptado a las limitaciones operativas del barco y evita el clima que puede causar la pérdida de carga.

SPOS Seakeeping es utilizado por compañías navieras que buscan reducir los costes de combustible y tiempo, aumentar la seguridad del barco y la tripulación, evitar el daño al medio ambiente y mejorar las relaciones con sus clientes. Algunas de las compañías que han implementado SPOS Seakeeping son TORM [28], una de las principales compañías de transporte de productos petrolíferos del mundo, o la gran empresa Maersk Line en 2013 [29]. Estas compañías usan SPOS Seakeeping porque les permite anticiparse a las condiciones meteorológicas adversas y planificar rutas que minimicen el riesgo de perder contenedores en el mar, lo que puede suponer un gran problema para la industria.

SPOS Seakeeping es una solución integrada de optimización de rutas marítimas que se basa en tecnologías innovadoras para mejorar el rendimiento de los barcos. Esta solución utiliza pronósticos meteorológicos actualizados varias veces al día para optimizar los parámetros de viaje. También utiliza modelado hidrodinámico para evaluar cómo el barco se manejará con la fuerza de las olas y planificación y ajuste de rutas para crear opciones de ruta basadas en restricciones de tiempo, coste o combustible.

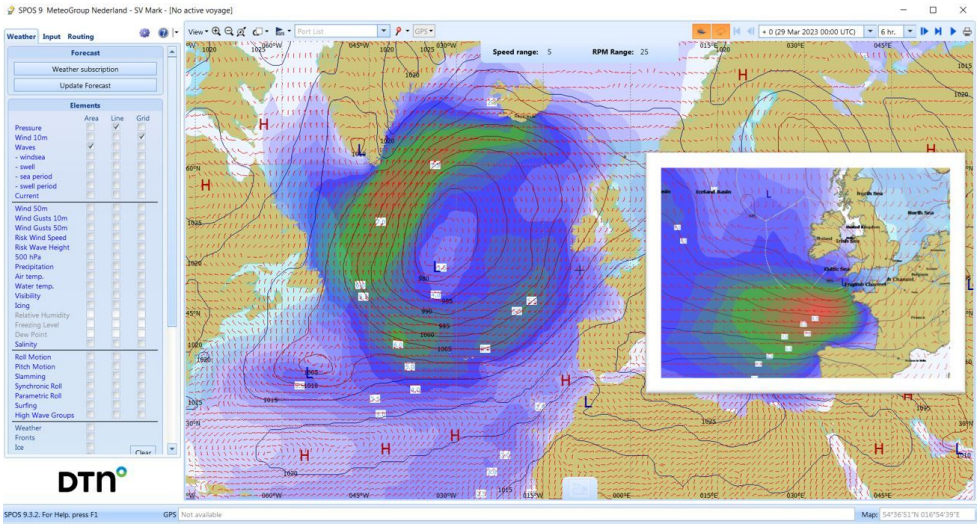

Figura 1.3: Plataforma de SPOS Seakeeping

SPOS Seakeeping tiene varios puntos fuertes que lo convierten en una herramienta valiosa para la industria naviera: SPOS optimiza las rutas de envío teniendo en cuenta las condiciones del mar, como el viento, las olas y el oleaje, las corrientes y otros elementos meteorológicos. Los capitanes pueden recibir actualizaciones meteorológicas oportunas y pronósticos para estar continuamente al tanto de su entorno y las condiciones que se avecinan. Esto permite a los capitanes navegar con seguridad por todo el mundo con el mínimo consumo de combustible y emisiones, calculando y recalculando las rutas óptimas y anticipándose a las condiciones meteorológicas y marítimas.

SPOS Seakeeping puede conectarse al sistema de asesoramiento marino ABB Ability^{TM} Marine Advisory System - OCTOPUS. El módulo es una integración completa en SPOS para ayudar a la tripulación a planificar su viaje teniendo en cuenta las respuestas y resonancias de los buques.

SPOS Seakeeping también es la única solución de seakeeping automatizada que incluye pronósticos meteorológicos y pronósticos de movimiento de buques en una interfaz de usuario. Funciona combinando pronósticos meteorológicos precisos y detallados con modelado hidrodinámico y datos operativos de cada buque para asesorar a los capitanes sobre la mejor ruta [30].

Sin embargo, el diseño sobrecargado y "anticuado" hace que a primera vista un usuario básico no sepa cómo poder navegar y conseguir sacar el máximo provecho de las funcionalidades. La cantidad de opciones y parámetros a configurar es alto.

Sin embargo, esta herramienta tiene un diseño complejo y "anticuado" que la hace menos competitiva y sostenible que otras soluciones disponibles en el mercado. Algunas de las desventajas de SPOS Seakeeping son: su interfaz de usuario poco intuitiva, su algoritmo de optimización de velocidad variable menos preciso y eficiente, su integración limitada con otros sistemas y plataformas, su dependencia tecnológica y costo adicional del sistema ABB Ability^{TM} Marine Advisory System – OCTOPUS y su alta cantidad de opciones y parámetros a configurar, lo que dificulta su uso y aprovechamiento por parte de los usuarios básicos.

1.4.3. Wayfinder

Wayfinder Voyage Optimization [31] es un sistema dinámico de orientación de viajes que ofrece las recomendaciones de velocidad y punto de paso más eficientes y menos restringidas por el clima a su flota. La plataforma se basa en su red global de sensores oceánicos, que producen los mejores pronósticos

meteorológicos marinos e informan modelos detallados de rendimiento de los buques (VPM). Wayfinder utiliza el estado físico actual del buque y sus pronósticos meteorológicos marinos para predecir con precisión la velocidad, el RPM y el consumo de combustible del buque para cualquier estado del mar encontrado. Wayfinder considera cientos de millones de opciones antes de entregar proactivamente la ruta más eficiente. Wayfinder también aprovecha los pronósticos meteorológicos marinos para proporcionar recomendaciones de seguridad oportunas, que limitan la exposición del buque al mal tiempo, previenen el desgaste estructural y protegen a la tripulación y la carga.

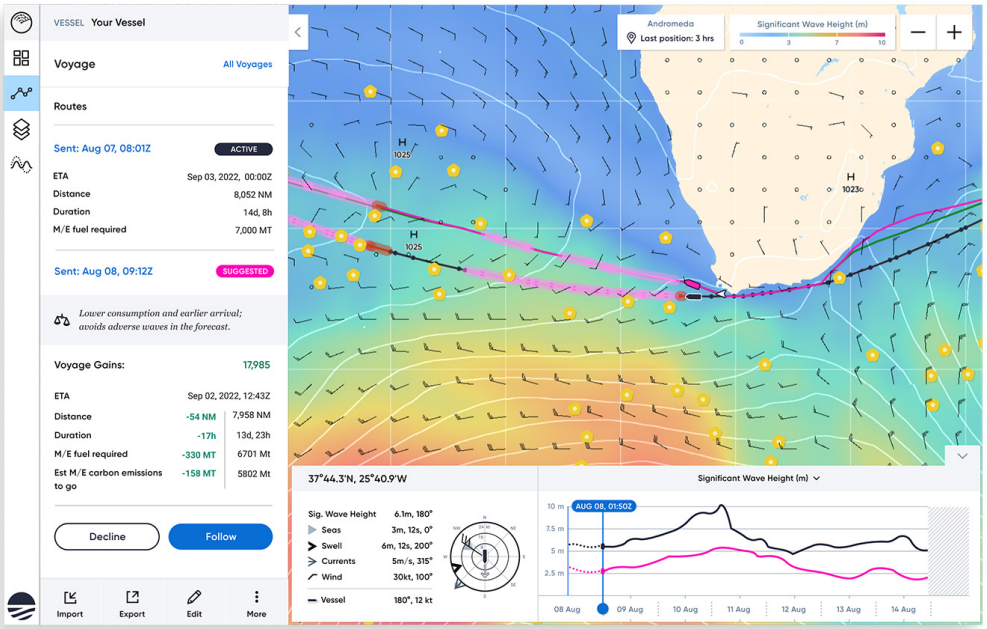

Figura 1.4: Plataforma de Wayfinder

Algunas compañías que utilizan Wayfinder Voyage Optimization son Berge Bulk, Star Bulk, ABS y Seaven. Estas compañías utilizan Wayfinder para aumentar la utilización y la eficiencia de los buques, asegurando que equilibran la rentabilidad con los objetivos de reducción de emisiones. Wayfinder les permite obtener ganancias de eficiencia en el rango del 4.5 % en viajes dedicados, lo que se traduce en hasta 14 días adicionales de navegación por año por buque. Wayfinder también se integra perfectamente en el flujo de trabajo del capitán y es fácil de usar.

Las tecnologías que utiliza Wayfinder Voyage Optimization son los sensores oceánicos Sofar Spotter, los modelos meteorológicos marinos Sofar Ocean Weather y los modelos de rendimiento de los buques (VPM). Los sensores oceánicos Sofar Spotter son boyas inteligentes que recogen datos en tiempo real sobre las olas, el viento, las corrientes y otros factores de resistencia del buque. Estos datos se asimilan en los modelos meteorológicos marinos Sofar Ocean Weather, que producen pronósticos que son un 40 % más precisos que los de los principales centros gubernamentales. Los modelos de rendimiento de los buques (VPM) son modelos digitales del buque que se recalibran dinámicamente con los datos reales de rendimiento y las observaciones meteorológicas en tiempo real [32].

Los puntos fuertes de Wayfinder Voyage Optimization son su precisión, su proactividad y su seguridad. Wayfinder ofrece las mejores recomendaciones de ruta basadas en los mejores pronósticos meteorológicos marinos, lo que reduce la incertidumbre y mejora los resultados. Wayfinder también anticipa automáticamente la necesidad de reducir la velocidad, acelerar o cambiar de rumbo, asegurando que el buque siempre se mantenga en el camino más eficiente hacia el puerto. Wayfinder también proporciona alertas oportunas para evitar el movimiento no deseado del buque, como el balanceo sincrónico, y capas ajustables de mapas meteorológicos y su red de boyas Spotter para proporcionar una situación marítima actualizada.

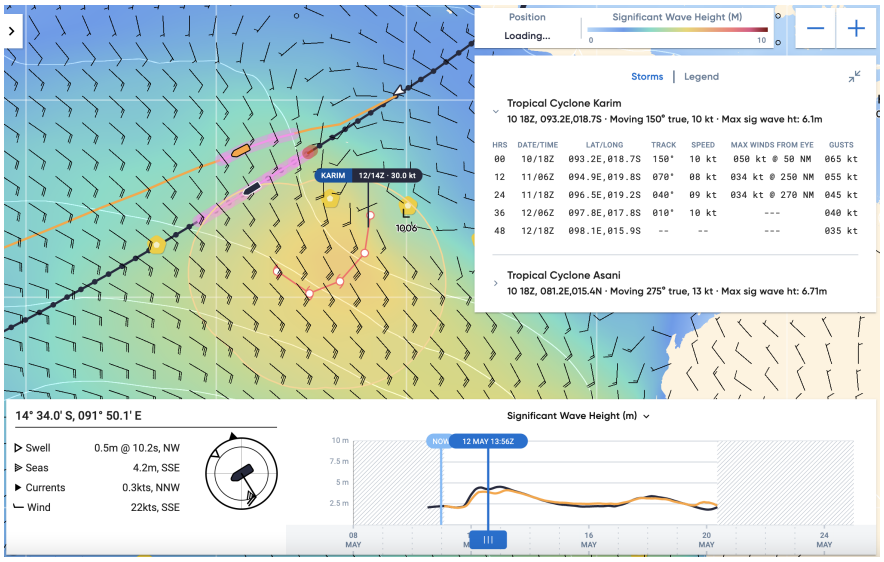

Figura 1.5: Plataforma de Wayfinder

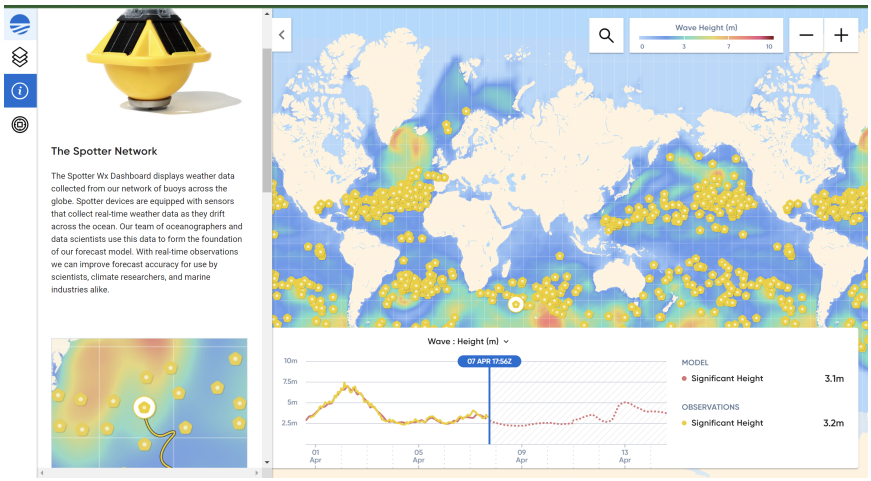

Figura 1.6: Sofar Spotter Weather Dashboard

Durante un viaje, Wayfinder también tiene en cuenta restricciones adicionales de viaje, como rangos de RPM prohibidos, hora requerida de llegada y asignación máxima diaria de combustible. Estas entradas adicionales aumentan la optimización en cuanto a la efectividad del motor, ya que puede ofrecer la ruta que tiene menos riesgos y más económica para cada buque según su situación particular [33].

No obstante, esta plataforma presenta algunas deficiencias y desafíos en comparación con otras soluciones existentes en el mercado, tales como: la falta de madurez y robustez de su desarrollo tecnológico, La limitada disponibilidad y suficiencia de la red global de sensores oceánicos Sofar en algunas regiones oceánicas de menor frecuencia o mayor complejidad de navegación, la omisión de los factores hidrodinámicos y las respuestas del casco en su modelo de rendimiento del buque y la escasa claridad sobre su integración con otros sistemas y plataformas.

1.4.4. SINAY

SINAY [34] es una solución de software que recopila, analiza y monitoriza datos marítimos complejos en una sola plataforma, utilizando diferentes módulos ambientales y logísticos. Su objetivo es ayudar a las industrias marítimas a aprovechar los datos y obtener información precisa y clave para mejorar

su eficiencia operativa, reducir costes y aumentar su competitividad, al mismo tiempo que reducen su impacto ambiental. Entre las compañías que utilizan SINAY se encuentran contratistas marinos, puertos inteligentes y compañías navieras. Algunas de las razones por las que eligen SINAY son: obtener predicciones precisas de ETA (tiempo estimado de llegada), optimizar las rutas de navegación, seguir el rastro de sus buques en tiempo real, monitorear la calidad del agua, del aire y del ruido, y acceder a datos metoceánicos.

SINAY utiliza tecnologías avanzadas de inteligencia artificial, análisis de datos y APIs para proporcionar sus servicios. También colabora con la Agencia Espacial Europea para crear una plataforma marítima digital que integre datos espaciales y terrestres.

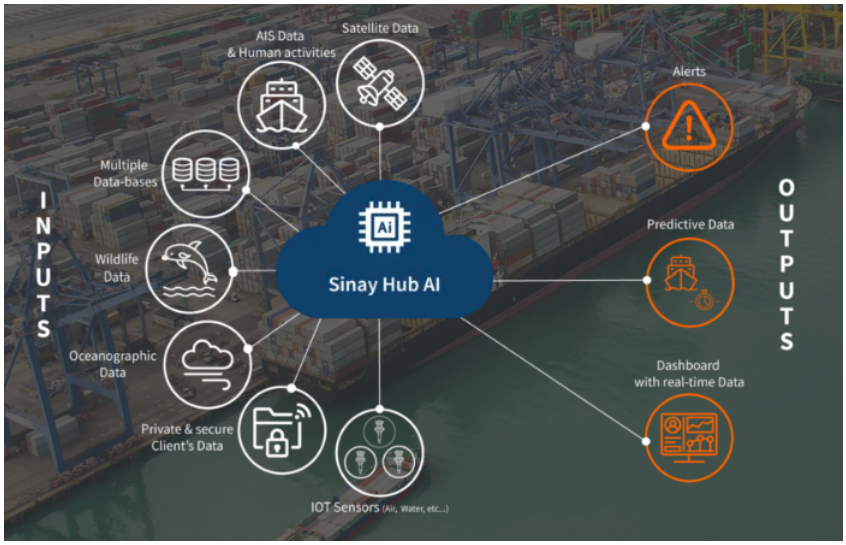

Figura 1.7: Sinay

SINAY es una solución que recolecta, analiza y supervisa datos marítimos complejos en una sola plataforma, ayudando a las empresas marítimas a mejorar la eficiencia de sus negocios y el seguimiento de su impacto ambiental. Esta tiene como puntos fuertes su capacidad para transformar datos marítimos complejos en indicadores simples para la toma de decisiones, su flexibilidad para adaptarse a las necesidades específicas de cada cliente, su innovación constante en el sector marítimo y su compromiso con la sostenibilidad. Sin embargo, SINAY también presenta algunos puntos débiles en comparación con otras soluciones disponibles en el mercado, tales como: su dependencia de la calidad y disponibilidad de los datos externos, su limitación a ciertas regiones geográficas, su competencia con otras soluciones similares en el mercado y su falta de especialización en la optimización de rutas marítimas basada en las condiciones climáticas y oceánicas, ya que su plataforma no ofrece orientación sobre la velocidad y el rumbo más eficientes para cada buque, sino que se limita a mostrar el tiempo estimado de llegada (ETA) y el consumo estimado de combustible [35].

Las principales conclusiones o implicaciones de nuestra comparación son que no existe un software óptimo para todos los casos, sino que cada uno tiene sus ventajas y desventajas dependiendo del contexto y las necesidades específicas.

1.5. Estructura de la memoria

En esta Sección se presenta la estructura y contenido seguida en los siguientes capítulos de la memoria del Trabajo de Fin de Grado:

- **Capítulo 1: Introducción.** En este Capítulo se ha llegado a dar una aproximación al proyecto en contexto del sector marítimo, la motivación de la realización del proyecto y los objetivos que se pretenden lograr con la realización de este proyecto, además de contar con un análisis de las soluciones existentes.
- **Capítulo 2: Planificación del proyecto.** A lo largo de este Capítulo se realizará una descripción de la metodología adoptada, alcance, plan de desarrollo del proyecto, plan de riesgos y su gestión, presupuesto inicial como un seguimiento del proyecto y las tareas llevadas a cabo.
- **Capítulo 3: Estado del Arte.** En este Capítulo se realizará un recorrido sobre el sector marítimo, en el que se engloba el concepto de ruta meteorológica, consumo de fuel y climatología; además se trata el problema desde un enfoque de optimización, contando con un apartado de investigación y trabajos relacionados. Además se presenta el concepto de Algoritmos Genéticos, en específico el NSGA-II y GA+PSO y se propone los modelos de machine learning para predecir la velocidad sobre el suelo (SOG).
- **Capítulo 4: Análisis.** En este Capítulo se profundizará en las diferentes etapas del análisis del proyecto: elicitación de requisitos en formato de historias de usuario, modelo de dominio del problema, los casos de uso y los respectivos diagramas de flujos de los algoritmos planteados.
- **Capítulo 5: Diseño.** En este Capítulo se describe la fase de diseño del proyecto, los patrones arquitectónicos seguidos, arquitectura del sistema y el diseño de la interfaz de usuario.
- **Capítulo 6: Modelo de Datos.** En este Capítulo se hace un recorrido completo del proceso de extracción, transformación y uso de los datos. Para ello primero se plantea el objetivo para la utilización de estos, se hace una descripción de los mismo y la modelización, es decir, la transformación de los datos originales al conjunto de datos final, que se utiliza para construir los modelos de machine learning predictivos del SOG.
- **Capítulo 7: Construcción del modelo.** En este Capítulo se describe el paso a paso que se ha realizado para la construcción del algoritmo NSGA-II. Desde cómo se ha utilizado el modelo de datos para modelar el espacio marítimo hasta la implementación del algoritmo genético.
- **Capítulo 8: Implementación.** En este Capítulo se describen los detalles relativos a la implementación del proyecto, más en concreto, el uso de las herramientas y librerías en las que nos hemos apoyado para llevarlo a cabo.
- **Capítulo 9: Pruebas.** En este Capítulo se exponen las pruebas realizadas para comprobar el buen funcionamiento del software desarrollado, además de mostrar diferentes escenarios meteorológicos para ver el comportamiento del algoritmo.
- **Capítulo 10: Conclusiones y Líneas Futuras.** En este último Capítulo se enuncias las conclusiones a la hora de la realización del proyecto como aquello puntos que se podrían mejorar como trabajo futuro.
- **Apéndices** En este apartado del documento se detalla la información adicional a la entrega del proyecto, las pautas como manuales de usuario e instalación para poder ejecutar el proyecto.

Capítulo 2

Planificación del proyecto

2.1. Metodología Scrum

En la planificación del proyecto software se ha elegido utilizar una metodología ágil [1] basada en Scrum. Esta metodología fue seleccionada debido a su capacidad de adaptación a los proyectos de desarrollo de software y su habilidad para cumplir con las exigencias óptimas en términos de flexibilidad, velocidad y eficiencia.

En el modelo Scrum[36], se adopta un enfoque ágil basado en la iteración y el incremento. Los proyectos se dividen en pequeñas partes de trabajo que se desarrollan y entregan gradualmente a lo largo de sprints de tiempo definido. Al final de cada sprint, se realiza una evaluación del progreso con la participación de los miembros del equipo y los interesados, permitiendo sugerir cambios y mejoras necesarias.

En el contexto de la metodología ágil, Scrum es un marco que se basa en la combinación de un enfoque iterativo y incremental con una perspectiva empírica. Esta metodología se caracteriza por su capacidad de adaptación a los cambios y por tres pilares clave: transparencia, inspección y adaptación.

La metodología Scrum es apta para proyectos que presentan un aspecto de investigación y/o una incertidumbre elevada, lo que significa que pueden estar sujetos a muchos cambios y necesitan retroalimentación constante debido a su naturaleza compleja [37].

Scrum cuenta con tres componentes clave para su implementación: **roles, eventos y artefactos**, que se describirán en los siguientes apartados.

2.1.1. Roles

El equipo de trabajo en Scrum es multifuncional, todos los miembros tiene la habilidad de crear valor durante cada ciclo de trabajo sin necesidad de apoyo externo, y autónomo en la toma de decisiones y asignación de tareas. Dentro del equipo de Scrum se definen tres responsabilidades específicas:

- **Product Owner.** Es el encargado de maximizar el valor del producto y de la gestión de la pila de producto (Product Backlog). Es responsable de desarrollar y comunicar el objetivo del producto, crear elementos de trabajo pendiente y asegurarse de que sea visible y comprensible. La organización debe respetar las decisiones del Product Owner, quien puede representar las necesidades de muchas partes interesadas.

[1] Más información: `https://agilemanifesto.org/principles.html`

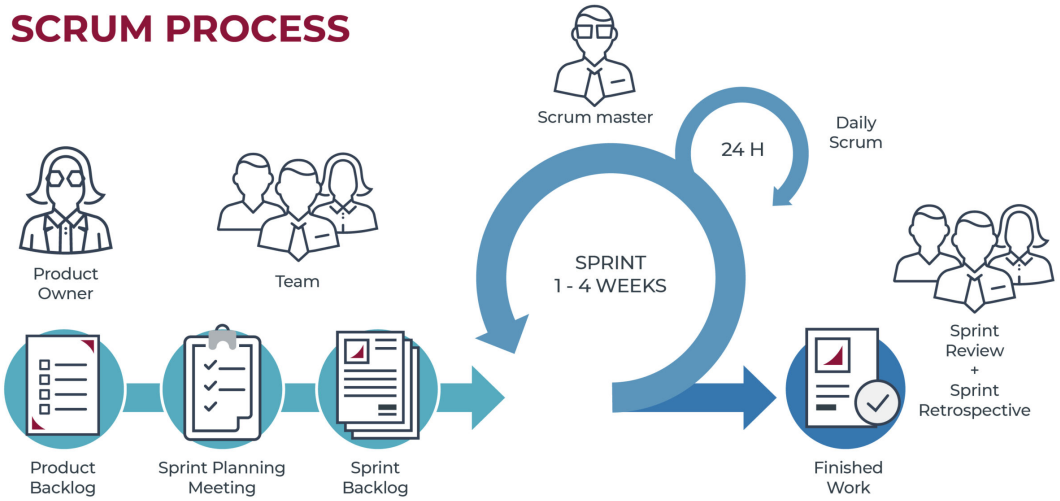

Figura 2.1: Framework Scrum

- **Scrum Master.** Es el líder encargado de ayudar en la comprensión y aplicación de Scrum dentro del equipo y la organización. Es responsable de mejorar la efectividad del equipo Scrum y ayudar en la gestión del producto, incluyendo la planificación empírica y la colaboración de las partes interesadas. El Scrum Master también lidera la adopción de Scrum en la organización, eliminando barreras entre equipos y partes interesadas.
- **Equipo de desarrollo.** Son miembros del equipo Scrum que se encargan de crear un incremento útil (funcional) durante cada Sprint. Deben poseer habilidades amplias y variadas, pero siempre deben planificar el Sprint, asegurarse de la calidad de su trabajo, adaptarse al objetivo de cada Sprint y ser responsables mutuamente como profesionales.

2.1.2. Eventos

Scrum utiliza eventos formales para inspeccionar y adaptar los artefactos, crear regularidad y minimizar la necesidad de reuniones no definidas. Estos eventos son importantes para permitir la transparencia necesaria en el proceso. Se recomienda que todos los eventos se lleven a cabo en el mismo lugar y tiempo para reducir la complejidad. Y estos son los siguientes:

- **Sprint.** Se trata del evento principal, del cuál giran alrededor el restos de eventos. Son ciclos de trabajo de un mes o menos de duración,que se desarrollan de manera secuencial a lo largo del proceso de desarrollo del proyecto. En los Sprints se realiza todo el trabajo necesario para alcanzar el objetivo del producto(incremento) y se llevan a cabo la planificación, daily scrum, revisión y retrospectiva. El objetivo del Sprint no se cambia durante el mismo, y se asegura la calidad y el refinar el trabajo pendiente en futuros Sprints. Cada Sprint puede considerarse un proyecto corto, y los Sprints más cortos ayudan a limitar el riesgo de coste y esfuerzo.
- **Sprint Planning.** Evento que inicia el Sprint, donde todos los componentes del equipo Scrum se reúnen para planear y priorizar el trabajo para el actual sprint. En dicha reunión, se eligen items del Product Backlog y se les da una estimación en puntos de historia para un tiempo de trabajo estimado. Cuanto más conozcan los desarrolladores sobre su desempeño previo, su capacidad futura y la claridad en sus objetivos, más precisas serán sus previsiones para el Sprint. Este evento da lugar al Sprint Backlog.

- **Daily Scrum.** Reunión diaria de 15 minutos en el que los miembros del equipo Scrum revisan su progreso y ajustan su plan para alcanzar el objetivo del Sprint. Es una oportunidad para mejorar la comunicación, identificar obstáculos y tomar decisiones rápidas. No es la única oportunidad para hacer ajustes y los miembros del equipo pueden reunirse adicionalmente durante el día para discutir temas más detallados.

- **Sprint Review.** Evento en el que el equipo de Scrum presenta los resultados del Sprint a las partes interesadas clave y se discute el progreso hacia el objetivo del producto. Durante el evento, se revisa el progreso y se ajusta el trabajo pendiente del producto en base a nuevas oportunidades, pudiéndose ajustar el Product Backlog.

- **Spring Retrospective.** Evento que concluye el Sprint, donde el equipo de Scrum evalúa este y planifica formas de mejorar la calidad y eficacia. Se identifican las suposiciones que los desviaron, se analiza lo que fue bien y lo que no, y se identifican cambios para mejorar la eficacia.

2.1.3. Artefactos

Son los resultados en forma de trabajo o valor por parte de las actividades en Scrum. Son herramientas clave para la transparencia y el progreso del equipo, y son los siguientes:

- **Product Backlog.** Se trata de una lista ordenada por importancia de características y mejoras para alcanzar el producto,representadas en la forma de historias de usuario. Estos elementos son refinados y definidos con más detalles antes de ser seleccionados en la planificación del Sprint. Los desarrolladores son responsables del tamaño y el Product Owner puede influir en ellos para mejorar la selección de tareas. El objetivo a alcanzar es el Product Goal, que describe un estado futuro del producto y se encuentra en el trabajo pendiente del producto, siendo un medio para entregar valor y cumplir antes de asumir el siguiente.

- **Sprint Backlog.** Es un plan realizado por los desarrolladores para alcanzar el objetivo Sprint durante el ciclo de trabajo. Está compuesto por el objetivo Sprint, los elementos de trabajo seleccionados y un plan accionable. Se actualiza a lo largo del Sprint y debe ser inspeccionable en el Daily Scrum. El Sprint Goal es el único objetivo para el Sprint y proporciona flexibilidad y enfoque para el equipo de Scrum. Es creado durante el evento Sprint Planning y es un compromiso para los desarrolladores.

- **Incremento.** Es la acumulación de los trabajos realizados en cada sprint, que juntos se aproximan al objetivo final del proyecto. El equipo de desarrollo es responsable de crear el incremento que debe cumplir con la Definición de Hecho (Definition of Done), estas condiciones determinan la utilización del mismo y será lo que se entregue al cliente al final del Sprint.

2.2. Adaptación Scrum al proyecto

El motivo de elegir Scrum para este proyecto es debido a su flexibilidad para adaptarse a los cambios que surjan, ya que partimos de una idea con unos requisitos no muy definidos, y su seguimiento constante semanal por parte de la tutora que permita retroalimentar e ir progresando en el proyecto. Además, Scrum es una de las formas de desarrollo ágil más utilizadas en el sector y permitirá aprender y poner en práctica una metodología muy demandada actualmente. Aunque se requiere una adaptación debido a limitaciones en personal, Scrum es un marco de referencia que se puede aplicar en diferentes contextos debido a su adaptabilidad [38].

En la adaptación que llevaremos a cabo, la tutora del proyecto, Margarita Gonzalo Tasis, toma el papel de Product Owner, facilitando la clasificación de las tareas por importancia y aportar retroalimentación sobre el desarrollo de cada Sprint. Por otra parte, al mismo tiempo el alumno satisfará las funciones de Scrum Master y equipo de desarrollo.

El cronograma del proyecto consiste en Sprints de 2 semanas cada uno. Los eventos clave de Sprint Planning, Sprint Review y Sprint Retrospective se llevarán a cabo cada 2 semanas, puntualmente en el final de un sprint y el comienzo del siguiente. Este día ha sido elegido el viernes. Las reuniones diarias de Scrum se han transformado en reuniones semanales, que se llevarán a cabo una vez por semana en los momentos en los que no se inicie ni finalice un sprint. En resumen, el viernes será el día en que el estudiante y la tutora se reúnan y, dependiendo del avance del sprint, la reunión tendrá un enfoque diferente.

En relación a los artefactos, el estudiante será el encargado de gestionar tanto el Product Backlog, el Sprint Backlog y el Incremento, ya que ha asumido los roles a los que quedaban a cargo.

2.3. Planificación Inicial

Para garantizar el éxito del proyecto, se ha creado un plan estratégico con un calendario de Sprints que incluye 9 etapas consecutivas. Cada Sprint tiene una duración fija de 2 semanas, pero el volumen de trabajo asociado puede variar. Con una dedicación semanal estimada de entre 15 y 20 horas, cada Sprint tendrá una duración aproximada de 30 a 40 horas, teniendo en cuenta otros compromisos académicos.

El plan de trabajo se ajusta en la reunión de Planificación de Sprint previa a cada Sprint, en la que se utilizan puntos de historia para estimar el tiempo de trabajo necesario. La guía docente[39] señala que la realización del Trabajo de Fin de Grado equivale a 12 ECTS o 300 horas, y esto se ha tenido en cuenta en la elaboración del plan.

El plan inicial incluye un sprint 0, el cual se distingue de los demás debido a su duración diferente. Este Sprint estará enfocado en tareas clave para asegurar la viabilidad del proyecto, como la investigación sobre el tema, la planificación detallada y la preparación del equipo. Con esto, el Sprint 1 estará listo para comenzar el desarrollo del proyecto de manera efectiva. Además, se han incorporado dos Sprints adicionales para brindar un margen de maniobra en caso de que no se complete todo el trabajo deseado en los 9 Sprints principales, y para realizar las últimas revisiones y mejoras en la documentación y material presentado.

El plan inicial incluye un total de 9 Sprints, con una duración promedio de 35 horas por sprint. A eso se suma el tiempo del sprint 0, que se estima en 20 horas, lo que nos da un total de 335 horas. En caso de que surjan imprevistos durante el proyecto, se han incluido dos Sprints extras de apoyo para garantizar que se cumpla con los objetivos planteados. Con estos Sprints adicionales, la estimación inicial queda en 370 o 405 horas, dependiendo de su uso.

En la tabla **2.1** se presenta una visión general de los Sprints y eventos previstos para la ejecución del proyecto.

Nº Sprint/Evento	Fecha Inicio	Fecha Fin	Anotaciones
Sprint 0	10/11/2022	27/01/2023	
Sprint Review, Sprint Retrospective y Sprint Planning	27/01/2023		
Sprint 1	27/01/2023	10/02/2023	
Sprint Weekly	03/02/2023		

Sprint Review, Sprint Retrospective y Sprint Planning	10/02/2023		
Sprint 2	10/02/2023	24/02/2023	
Sprint Weekly	17/02/2023		
Sprint Review, Sprint Retrospective y Sprint Planning	24/02/2023		
Sprint 3	24/02/2023	10/03/2023	
Sprint Weekly	03/03/2023		
Sprint Review, Sprint Retrospective y Sprint Planning	10/03/2023		
Sprint 4	10/03/2023	24/03/2023	
Sprint Weekly	17/03/2023		
Sprint Review, Sprint Retrospective y Sprint Planning	24/03/2023		
Sprint 5	24/03/2023	07/04/2023	
Sprint Weekly	31/03/2023		Vacaciones Semana Santa
Sprint Review, Sprint Retrospective y Sprint Planning	07/04/2023		Vacaciones Semana Santa
Sprint 6	07/04/2023	21/04/2023	
Sprint Weekly	14/04/2023		
Sprint Review, Sprint Retrospective y Sprint Planning	21/04/2023		
Sprint 7	21/04/2023	05/05/2023	
Sprint Weekly	28/04/2023		
Sprint Review, Sprint Retrospective y Sprint Planning	05/05/2023		
Sprint 8	05/05/2023	19/05/2023	
Sprint Weekly	12/05/2023		
Sprint Review, Sprint Retrospective y Sprint Planning	19/05/2023		
Sprint 9	19/05/2023	02/06/2023	
Sprint Weekly	26/05/2023		
Sprint Review y Sprint Retrospective	02/06/2023		También Sprint Planning si se necesita para desarrollo el Sprint siguiente.
Sprint extra 1	02/06/2023	16/06/2023	
Sprint Weekly	09/06/2023		
Sprint Review y Sprint Retrospective	16/06/2023		También Sprint Planning si se necesita para desarrollo el Sprint siguiente.
Sprint extra 2	16/06/2023	30/06/2023	
Sprint Weekly	23/06/2023		
Límite solicitud defensa convocatoria ordinaria	23/06/2023		Tribunales antes del **05/07/2023**
Sprint Review, Sprint Retrospective y Sprint Planning	30/06/2023		Se podría adelantar si se decide entregar en convocatoria ordinaria.

Revisión documentación	30/06/2023	10/07/2023	
Límite solicitud defensa convocatoria extraordinaria	11/07/2023		Tribunales antes del **20/07/2023**

Tabla 2.1: Planificación inicial de Sprints

2.4. Plan de riesgos

Durante el proceso de planificación, se creó un plan integral para enfrentar los desafíos y obstáculos que puedan surgir. Es comprensible que cualquier proyecto tenga un margen de incertidumbre, ya que se basa en suposiciones y pueden presentarse situaciones imprevistas que alteren esas predicciones.

Un riesgo[40] es una situación potencial que puede tener un impacto positivo (oportunidad) o negativo (amenaza) en el futuro del proyecto con una causa y un efecto. Al evaluar los riesgos, es importante tener en cuenta su probabilidad de ocurrencia y su potencial impacto en el desarrollo del proyecto. Para mitigar el efecto negativo de estos riesgos, se requiere un plan estratégico de manejo de riesgos, que permita prevenir y controlar la aparición de situaciones inesperadas y minimizar sus efectos en el progreso del proyecto.

En este plan de manejo de riesgos, se han considerado cuatro elementos para cada riesgo identificado:

- **Probabilidad.** Refiere a la posibilidad de que el riesgo suceda y se clasifica en baja, media o alta.
- **Impacto.** Refiere al alcance del efecto que tendría el riesgo si se materializa y se evalúa como bajo, medio o alto.
- **Plan de mitigación.** Son las acciones encaminadas a disminuir la posibilidad de que el riesgo se concrete.
- **Plan de contingencia.** Son las medidas que se pueden implementar una vez que el riesgo se manifiesta, para minimizar su impacto.

El plan de manejo de riesgos es la clave para garantizar el éxito del proyecto, ya que permite identificar y prevenir los desafíos que puedan surgir. Con un enfoque en la prevención y la mitigación, se establecen acciones tanto para antes como para después de la materialización de cada riesgo, de manera que se esté preparado en todo momento. Los riesgos se han clasificado en dos categorías: riesgos generales, que pueden ocurrir en cualquier contexto, y riesgos particulares, que tienen en cuenta las características específicas del proyecto en cuestión.

Se pueden observar los riesgos generales en las Tablas **2.3** a **2.8**, y los particulares de la Tablas **2.9** a **2.10**.

Riesgo R01	**Falta de disponibilidad**
Descripción	Debido a la larga duración del proyecto, es posible que la disponibilidad del equipo de desarrollo varíe, lo que podría requerir cambios en la planificación de los sprints para ajustarse a las nuevas circunstancias.
Probabilidad	Media
Impacto	Medio
Plan de mitigación	▪ Mantener un equilibrio entre su trabajo, su vida personal y el proyecto para evitar un agotamiento o estrés. ▪ Evitar actividades de riesgo o con posibilidad de accidente, a la vez que mantener una dieta equilibrada y cuidar su salud.
Plan de contingencia	▪ Distribuir las tareas no realizadas en siguientes Sprints, asignando prioridades, para asegurar la completitud de las actividades más importantes. ▪ Planificar con flexibilidad y dejar margen para la fecha de entrega. ▪ Considerar la posibilidad de reducir la jornada laboral o tomar vacaciones para dedicar más tiempo al proyecto.

Tabla 2.2: Riesgo R01

Riesgo R02	**Fallos técnicos en el equipo de trabajo del estudiante**
Descripción	El equipo de trabajo del estudiante puede experimentar fallos técnicos, como problemas de hardware o software, lo que puede obstaculizar el progreso del proyecto y retrasar su finalización.
Probabilidad	Baja
Impacto	Medio
Plan de mitigación	▪ Hacer copias de seguridad regulares de los documentos importantes ▪ Almacenarlos en un repositorio/nube online para protegerlos de posibles fallos técnicos. ▪ Trabajar en la medida de lo posible con herramientas online para reducir la dependencia del equipo local con herramientas como Overleaf o Google Colab.
Plan de contingencia	▪ Recuperar la información, documentos o código desde otros dispositivos, si se han realizado copias de seguridad. ▪ A través de los mecanismos de recuperación por parte de los programas utilizados, intentar recuperar los archivos.

Tabla 2.3: Riesgo R02

Riesgo R03	**Dificultades en la comunicación del equipo por limitaciones de horario**
Descripción	Debido a los compromisos personales o académicos de los miembros del equipo, puede haber momentos en los que la comunicación sea difícil o incluso imposible, lo que podría retrasar el progreso del proyecto y afectar su planificación.
Probabilidad	Media
Impacto	Bajo
Plan de mitigación	▪ Establecer un horario regular de reuniones y utilizar herramientas de comunicación como correo electrónico y chat para mantenerse en contacto. ▪ Fomentar la comunicación clara y efectiva entre los miembros del equipo para minimizar malentendidos.
Plan de contingencia	▪ En caso de que la comunicación sea difícil, se buscarán alternativas como posponer reuniones o asignar tareas que no requieran la participación/supervisión directa de todos los miembros del equipo ▪ Establecer un plan de recuperación de tiempo para evitar retrasos en el proyecto.

Tabla 2.4: Riesgo R03

Riesgo R04	**Generación incorrecta de artefactos en la metodología propuesta**
Descripción	Los artefactos desarrollados durante el proyecto pueden no cumplir con las expectativas y requisitos de los miembros del equipo, lo que puede llevar a tener que rehacer ciertos trabajos o a que la memoria final no cumpla con los objetivos propuestos.
Probabilidad	Media
Impacto	Medio
Plan de mitigación	▪ Realizar una documentación detallada de la metodología a seguir y de los artefactos que deben generarse, con el fin de evitar errores y malentendidos. ▪ Comunicación del desarrollo y evolución de los artefactos al resto del equipo y que se aprueben en conjunto.
Plan de contingencia	▪ En caso de detectar que los artefactos generados no cumplen con los requisitos, se realizarán reuniones internas para detectar los problemas y se tomarán medidas correctoras para solucionarlos en el menor tiempo posible. ▪ En caso de ser necesario, se reestructurará la planificación del proyecto para poder seguir las indicaciones necesarias.

Tabla 2.5: Riesgo R04

Riesgo R05	**Error en la estimación de tiempo de las actividades a desarrollar**
Descripción	La estimación de tiempo y esfuerzo para algunas actividades o tareas planificadas resulta ser incorrecta y no se ajusta a la realidad, lo que provoca un retraso en la finalización del trabajo planificado.
Probabilidad	Media
Impacto	Medio
Plan de mitigación	▪ Realizar un análisis de cada tarea y actividad en profundidad para obtener una estimación más precisa. ▪ Posponer la atención en los aspectos menores del proyecto y enfocarse principalmente en las partes críticas del mismo hasta su culminación. ▪ Contar con Sprints extras en la planificación de desarrollo para cubrir aquellos aspectos que no se han cumplido
Plan de contingencia	▪ Usar esos Sprint extras para completar el proyecto. ▪ Reducir el alcance del proyecto en la medida en que resulte factible.

Tabla 2.6: Riesgo R05

Riesgo R06	**Falta de familiaridad con la metodología Scrum**
Descripción	El equipo encargado de llevar a cabo el proyecto no tiene experiencia previa con la metodología Scrum. Esto puede resultar en una planificación inadecuada, priorización incorrecta de tareas y problemas para definir los roles adecuados en el equipo.
Probabilidad	Media
Impacto	Medio
Plan de mitigación	▪ Se llevará a cabo una capacitación exhaustiva en los principios y prácticas de Scrum, haciendo énfasis en la revisión constante para asegurar la correcta aplicación de la metodología.
Plan de contingencia	▪ Si alguno de los integrantes del equipo no cumple con su tarea de forma adecuada, se colaborará en conjunto para cubrir esa falta y ajustarse para el siguiente Sprint de trabajo.

Tabla 2.7: Riesgo R06

Riesgo R07	**Falta de conocimiento de las tecnologías y herramientas necesarias para el proyecto**
Descripción	El estudiante no está familiarizado con las herramientas y tecnologías que son necesarias para llevar a cabo el proyecto, lo que puede provocar dificultades en el proceso de desarrollo.
Probabilidad	Media
Impacto	Medio
Plan de mitigación	▪ Adquirir conocimiento previo de las herramientas y tecnologías necesarias antes de empezar el proyecto, a través de cursos, tutoriales y documentación.
Plan de contingencia	▪ Busca alternativas factibles y ya conocidas para utilizar en lugar de la nueva herramienta. ▪ Observar documentación oficial o tutoriales que expliquen el manejo de dichas herramientas. ▪ Si conocimiento adquirido no es suficiente, se buscará asesoría o ayuda de expertos en el área.

Tabla 2.8: Riesgo R07

Riesgo R08	**Cambios en la estructura de las fuentes de datos**
Descripción	Las fuentes de datos utilizadas en el proyecto pueden experimentar cambios en su estructura, lo que puede ocasionar fallos en su extracción y actualización.
Probabilidad	Media
Impacto	Alto
Plan de mitigación	▪ Identificar las fuentes de datos críticas y establecer un proceso de seguimiento y monitoreo para detectar cambios en su estructura.
Plan de contingencia	▪ Realizar ajustes en el proceso de extracción para asegurar la actualización de los datos afectados. ▪ Considerar la búsqueda de nuevas fuentes de datos alternativas en caso de que no se puedan extraer los datos necesarios de la fuente original.

Tabla 2.9: Riesgo R08

Riesgo R09	**Modificación de los requisitos del proyecto**
Descripción	Durante el desarrollo del proyecto, pueden surgir cambios en los requisitos que afecten la planificación inicial y produzcan modificaciones en el alcance final.
Probabilidad	Media
Impacto	Medio
Plan de mitigación	▪ Es importante definir y priorizar desde el inicio las funcionalidades importantes y críticas, para evitar cambios continuos en los requisitos y mantener la estabilidad en el proyecto. ▪ Realizar entregas frecuentes del proyecto y obtener retroalimentación de los usuarios, incluyendo la tutora, para validar el progreso y hacer ajustes o mejoras según sea necesario.
Plan de contingencia	▪ Si es un requisito crítico, hacer uso de los Sprint extras para desarrollarlo. ▪ Si no es crítico, dejarlo fuera de la realización del proyecto y comentarlo como trabajo futuro.

Tabla 2.10: Riesgo R09

2.5. Plan de presupuesto

Como última etapa de la planificación, un plan presupuestario es una parte crítica de cualquier proyecto, por lo que se requiere un análisis exhaustivo de los costos involucrados en diferentes áreas, como el hardware, el software, los recursos humanos y la infraestructura. Aunque hay otros costos como el de la electricidad, estos no se consideran importantes en comparación con el presupuesto general.

Es necesario extrapolar los costos a lo largo del tiempo total del proyecto, que en este caso abarca un período de 5 meses, desde mediados de enero hasta principios de julio de 2023.

2.5.1. Coste del hardware

El costo de los componentes de hardware estima el valor de los diversos dispositivos electrónicos empleados en el proyecto. Los dispositivos utilizados han sido los siguientes:

- Ordenador portátil Asus Vivobook S14 (16GB de RAM y procesador i7 de 12º generación): tiene un coste de 999,00€ según la tienda oficial de Asus. Si consideramos una vida útil de 4 años (48 meses), y como el proyecto abarca 5 meses, el coste a contabilizar es de 104€[41].
- Monitor Samsung QHD 32"que tiene un coste de 299,00€ y una vida media de 10 años (120 meses). Por lo que, el coste al proyecto es de 12,50€[42].

El coste total del hardware, por tanto, es de 116,50€.

2.5.2. Coste del software

En este proyecto, se utilizaron programas de software completamente gratuitos debido a que se es estudiante. La universidad proporcionó licencias gratuitas para algunos programas, lo cual permitió aprovechar esta ventaja. Sin embargo, si se trasladara este proyecto a un entorno empresarial, donde los programas tienen un costo y existen múltiples tipos de licencias o ediciones para cada programa, el costo del software aumentaría en función de la opción elegida.

2.5.3. Coste de recursos humanos

En el contexto actual, es decir un Trabajo Fin de Grado, no se aplicaría un coste de recursos humanos. Sin embargo, en un entorno empresarial sí existiría un coste para este proyecto. Se estableció un plazo de 300 horas para la realización según la guía docente de la asignatura. Dado que se trata de un desarrollador con perfil junior, el salario medio por hora es de 13,54€ bruto en España, según Glassdoor [2]. Por lo tanto, el coste estimado sería de 4062,50€.

2.5.4. Coste de infraestructura

El estudiante ha llevado a cabo la mayor parte del trabajo para este proyecto en un piso de alquiler, y se incluirán en el presupuesto los gastos de suministros correspondientes.

[2] https://www.glassdoor.es/Sueldos/espa%C3%B1a-junior-data-scientist-sueldo-SRCH_IL.0,6_IN219_KO7,28.htm?clickSource=searchBtn

Para este presupuesto se estima un precio medio de 300€ al mes para el alquiler de la vivienda. Por lo que el coste prorrateado por las 335 horas planificadas, el coste total sería de unos 420€. Este precio incluye los gastos relacionados con la luz, el agua y el Internet.

2.5.5. Coste total

Para calcular el coste total del proyecto, se sumaron los valores de los costes anteriores y se añadió un 25 % extra como margen de seguridad. La Tabla **2.11** muestra los costes detallados anteriormente y la suma final del proyecto.

Concepto	**Coste**
Coste de hardware	116,50€
Coste de software	No aplicable (0€)
Coste de recursos humanos	4062,50€
Coste de infraestructura	420€
Coste total	**4600€**
Coste total + extra	**5750€**

Tabla 2.11: Costes presupuestados.

2.6. Seguimiento de los Sprints realizados

En esta Sección se presentarán el seguimiento del proyecto en cada uno de los sprints realizados. Cada uno de ellos se definirá las fechas en las que se llevó a cabo, las tareas asignadas a este , su completitud y el número de horas dedicas a este. Además se comentarán los riesgos que han sucedido durante el desarrollo de estos y a lo largo del proyecto.

Sprint 0

Este Sprint duró desde el 10/11/2022 hasta el 03/01/2023 cuyo objetivo era tratar la viabilidad del proyecto, posibilidades técnicas y de datos para poder llevarlo a cabo, y una investigación de los estudios realizados que abordaran el tema.

Por lo tanto con esta investigación, lo que se pretendía era llegar con el mayor conocimiento del contexto del proyecto para poder encarar el inicio sabiendo de las posibilidades técnicas disponibles para realizarlo.

Para llevarlo a cabo se emplearon **15 horas**.

Sprint 1

Este sprint tuvo lugar desde el 03/01/2023 hasta el 17/02/2023. En este primer sprint el objetivo fue fijar los límites al proyecto con toda la información recopilada en el Sprint 0 y tejer un plan de acción. Entre el resto de tareas tangibles a desarrollar, se destacan las siguientes entorno a la memoria técnica:

- Metodología.

- Planificación.
- Título y descripción.
- Riesgos y gestión.

Todas estas tareas fueron llevadas a cabo en el tiempo del sprint con una revisión y ajustes en la Sprint Weekly del 10/02/2023. Por último, en la Sprint Review del día 17, se acordó repasar la parte de Riesgos ya que no estaba del todo bien explicada y realizar la parte de Presupuesto.

Para ello se emplearon **30 horas**.

Sprint 2

Este sprint duró desde el 17/02/2023 hasta el 03/03/2023. En este sprint se fijaron completar y realizar las modificaciones del sprint anterior, como un diseño de la interfaz de usuario y una estructura de cómo quedaría la página web, además de probar diversas formas de acceder a los datos alojados en los servidores de Copernicus.

Por tanto las tareas se describen:

- Finalizar Riesgos.
- Finalizar Presupuesto.
- Bocetado y diseño de la UI de la página web.
- Aproximación a la extracción de los datos meteorológicos de Copernicus.

Estos objetivos y tareas se llevaron a cabo en su totalidad y se emplearon **30 horas**.

Sprint 3

Este sprint abarcó desde el 03/03/2023 al 17/03/2023 y en este periodo nos centramos en la parte de Análisis del proyecto, cuyas tareas podemos ver desglosadas en lo siguiente:

- Análisis de requisitos.
- Creación Historias de usuario.
- Diagramas y descripción casos de uso.
- Modelo de dominio.

Durante la Sprint Weekly y Sprint Review se fue iterando y refinando sobre el trabajo hecho para dejarlo completo de forma correcta. Para ello se emplearon **30 horas**.

Sprint 4

Este sprint llegó desde el 17/03/2023 al 31/03/2023 y en este se profundizó en la parte de Análisis que faltaba y en la búsqueda de información sobre Aplicaciones Similares:

- Diagramas de secuencia en análisis.
- Diagramas de flujo.
- Aplicaciones similares.

Cuando se planificó este sprint en la Sprint Planning del 17/03/2023, debido a una cierta falta de disponibilidad; ya que en el trabajo que desarrolla el estudiante tenían alta carga de trabajo, se decidió no sobrecargar de tareas este sprint. Contando con ello, las tareas anteriores se llevaron a cabo en su totalidad, exceptuando la de Aplicaciones Similares que no se completó en su totalidad. En total se emplearon **25 horas**.

Sprint 5

Este sprint duró desde el 31/03/2023 al 14/04/2023, y debido a que coincidieron con las Vacaciones de Semana Santa, se ideó terminar lo que faltaba del anterior Sprint, desarrollar un script para la descarga de datos y el aprendizaje sobre Flask. Por tanto las tareas fueron:

- Finalizar Aplicaciones similares.
- Creación Script de datos Copernicus y AIS.
- Aprendizaje y documentación sobre la herramienta Flask.

Debido a las vacaciones de Semana Santa, la Sprint Weekly no fue llevada a cabo debido a la falta de disponibilidad de la tutora por vacaciones ni la Sprint Review, Sprint Retrospective y Sprint Planning de este sprint debido, de nuevo, a la falta de disponibilidad por enfermedad de la tutora.

En cuanto a las tareas realizadas, apareció el riesgo previsto de cambios en la estructura de las fuentes de datos en la extracción de estos del servidor Copernicus. Desde la aproximación en el sprint 2 para conocer cómo acceder a la API, realizaron diversos cambios en la estructura y forma de acceso a los datos, lo que retraso la implementación y desarrollo del script de descarga de datos

A pesar de esto, se completaron las tareas asignadas, y mediante comunicación por correo electrónico a la tutora se comentaron los avances en el sprint y se fijaron la nueva asignación de tareas para el siguiente sprint. Por lo que en este sprint se dedicaron **35 horas**.

Sprint 6

Este sprint abarcó desde el 14/04/2023 al 28/04/2023. En este sprint, se solicitó la máquina virtual a la Escuela para poder llevar a cabo la descarga y unión de los conjuntos de datos del AIS, mostrar los avances de sprints anteriores para tener una foto fija del avance del proyecto e iniciar el Capítulo de Introducción en la memoria del proyecto:

- Revisión del proyecto.
- Puesta en funcionamiento script extracción de datos.
- Capítulo Introducción Memoria.

En cuanto a la tarea de poner en marcha el script de extracción de los datos, a mitad del proceso, Copernicus actualizó la estructura de los datos de uno de los dos conjuntos de datos, haciendo que para lograr la extracción de los datos necesarios, anteriormente se necesitaba una llamada; y con la nueva estructura se necesitan cuatro llamadas. Esta aparición del riesgo nos llevó a actualizar, de nuevo, el script para adaptarlo a las cuatro llamadas.

Esto nos llevó a completar todas estas tareas en el sprint en unas **30 horas**.

Sprint 7

Este sprint se llevó a cabo durante el 28/04/2023 hasta el 12/05/2023 con el objetivo de modelar el grid y el modelo de ML para predecir la velocidad SOG, a la vez de comenzar el Capítulo de Estado del Arte. Por consiguiente las tareas:

- Construcción Modelo de Datos.
- Documentación de la memoria del Estado del Arte.

La tarea relativa a la construcción del modelo de datos quedó completa, y del Estado del Arte quedó la parte relativa a algoritmo NSGA-II y los modelos de ML. Para la realización de estas tareas supuso unas **40 horas**.

Sprint 8

Este sprint abarcó desde el 12/05/2023 al 26/05/2023. Este sprint estuvo marcado por la finalización de la parte del Estado del Arte y la realización de la parte de documentación de diseño y arquitectura del sistema, por lo que el desglose de tareas queda:

- Finalización documentación Estado del Arte.
- Documentación de la memoria de diseño: Arquitectura, patrones de diseño, ...

De la documentación de diseño se fue refinando en la Sprint Weekly y Sprint Review, quedando algún detalle por completar para el siguiente sprint. Se emplearon unas **30 horas**.

Sprint 9

Este sprint duró desde el 26/05/2023 al 09/06/2023, y en este nos centramos en la finalización de la parte de diseño y en la implementación del algoritmo NSGA-II con la librería Pymoo. Para todo ello las tareas quedaron de la siguiente forma:

- Finalización documentación Diseño.
- Documentación Pymoo.
- Implementación NSGA-II.

De este sprint se completaron todas las tareas, exceptuando la implementación completa del algoritmo. Por lo que la implementación del cruce y mutación quedaron al siguiente sprint. Para llevar estas tareas a cabo se emplearon **35 horas**.

En la Sprint Weekly de este sprint se toma la decisión de la sola implementación del Algoritmo NSGA-II, ya que para los diferentes riesgos que han ido apareciendo hacen que tengamos mucho menos tiempo para el desarrollo, y por tanto sería inviable la implementación, además, del GA+PSO.

Sprint extra 1

Este sprint tuvo lugar desde el 09/06/2023 al 23/06/2023. En este nos centramos en la finalización del algoritmo, la conexión con Flask, la visualización de las rutas con Plotly.js y la documentación de la memoria de la implementación, quedando del siguiente modo la lista de tareas:

- Finalización algoritmo: Cruce y mutación.
- Documentación Plotly.js e implementación.
- Conexión con Flask.
- Documentación memoria Capítulo Implementación.

Quedaron completas todas las tareas en este sprint con un uso de unas **40 horas**.

Sprint extra 2

Este sprint se llevó a cabo desde el 23/06/2023 al 07/07/2023. Este último sprint se fijó para terminar la parte de la memoria y hacer una revisión del proyecto, y completar los últimos detalles de la parte técnica del proyecto. Por tanto las tareas fueron:

- Documentación memoria Capítulo Pruebas.
- Documentación memoria Capítulo Conclusiones y Líneas Futuras.
- Documentación memoria manual de usuario e instalación.
- Revisión Documentación memoria.
- Revisión código del proyecto.

Todas estas tareas fueron terminadas con éxito y se emplearon unas **40 horas**.

Conclusiones del desarrollo del proyecto

Como se ha comentado anteriormente durante el proceso de desarrollo han ido apareciendo diferentes riesgos que han retrasado la planificación inicial prevista, entre ellos: *Falta de disponibilidad*, *Falta de conocimiento de las tecnologías y herramientas necesarias para el proyecto* y *Cambios en la estructura de las fuentes de datos*.

Los dos primeros riesgos tuvieron un impacto medio a la hora de la consecución del proyecto, ya que la *Falta de disponibilidad* en el caso del alumno se tradujo en un número menor de horas dedicadas en ese sprint y *Falta de conocimiento de las tecnologías y herramientas necesarias para el proyecto* hizo que el aprendizaje y empleo de herramientas como *Flask*, *Pymoo* o *Plotly.js* llevaran más tiempo del inicialmente planteado.

Por último, mayor impacto por el tipo de riesgo y el número de apariciones, dio lugar el de *Cambios en la estructura de las fuentes de datos* ya que nos llevó a evaluar los errores y cambios a realizar, estudiar la nueva estructura propuesta por Copernicus y adaptar y modificar los scripts de descarga de datos; lo que nos retrasó la planificación.

Por todo ello, en el Sprint 9 se terminó finalmente de descartar la implementación del Algoritmo GA+PSO, ya que iba a ser el que más tiempo nos iba a llevar implementar de los dos, por lo que quedará como trabajo a futuro.

Haciendo una recapitulación por el conjunto de sprints, el sumatorio de horas empleadas para completar satisfactoriamente el proyecto se emplearon **380 horas**. Debido a que se emplearon finalmente los dos sprint extras y los retrasos producidos por los riesgos que surgieron la cifra llegó a ese número, dentro de lo planificado por el uso de los sprint extras.

Capítulo 3

Estado del arte

3.1. Estudio del Dominio: Sector Marítimo

La navegación marítima es una actividad única y tiene características que la diferencian de otros medios de transporte, como el movimiento de vehículos terrestres o aéreos (Tu et al., 2018; Ueno et al., 2009) [43] [44]. Se pueden identificar tres principales diferencias: en primer lugar, los barcos no pueden detenerse, girar o invertir su dirección de forma repentina, como sí lo pueden hacer los vehículos terrestres. En segundo lugar, los barcos se mueven en un plano horizontal, mientras que los aviones o submarinos lo hacen en un espacio tridimensional. Y en tercer lugar, el cambio de rumbo de un barco es costoso en términos de tiempo y consumo de combustible, lo que lleva a los barcos a minimizar los cambios de rumbo (Montewka et al., 2017)[45].

Aunque los barcos típicamente tienen una gran libertad de movimiento en alta mar, su movimiento puede ser restringido por varios factores, como aguas someras (aguas poco profundas), zonas de seguridad, canales de navegación, hielo y la capacidad de los puertos (Guinness et al., 2014; Lai y Shih, 1992; Löptinen y Axell, 2014; Seong et al., 2011)[46][47][48].

En las primeras etapas, el propósito principal de un servicio/sistema de optimización de viajes era proporcionar una guía principal a los barcos para llegar a su destino lo más rápido posible, basándose en las condiciones climáticas pronosticadas y posiblemente también en las características y capacidades operativas del barco (Bowditch, 2002)[49]. A medida que el precio del petróleo y la conciencia social sobre las emisiones atmosféricas de la industria naviera han aumentado rápidamente, los sistemas de optimización de viajes pueden proporcionar actualmente rutas óptimas para los viajes por océano que tengan en cuenta dicho factor.

Debido a las grandes incertidumbres (por ejemplo, inversión y de amortización) que a menudo conlleva la implementación de medidas complejas y costosas, las empresas navieras prefieren invertir en medidas con técnicas de implementación simples y soluciones rentables [50]. Entre todas las medidas disponibles, la planificación de la ruta teniendo en cuenta la meteorología es una de las más reconocidas implementaciones en el mercado. Los beneficios de utilizar estos sistemas de enrutamiento están bien documentados en (Chen, Cardone, & Lacey,1998; Buhaug et al.,2009) [51][52]

Mientras tanto, un factor importante en la competitividad dentro de la industria naviera es la minimización del consumo de combustible. Esto se puede lograr en parte mediante una ruta óptima, que debe tener en cuenta simultáneamente la seguridad de la carga y del personal, y el tiempo de llegada, entre otros.

Las rutas meteorológicas de los buques son el desarrollo de un rumbo y una velocidad de navegación

óptimos para los viajes oceánicos, basado en cartas náuticas, las condiciones marítimas previstas y posiblemente las características individuales de un buque para un tránsito específico [49]. Tanto el consumo de combustible como el tiempo de llegada esperado están vinculados a la ruta óptima, pero también están directamente conectados con los perfiles de velocidad/potencia que desarrolla el barco [53]. Por lo tanto, la mayoría de los buques de cruce oceánico están instrumentados con un sistema de enrutamiento meteorológico para planificar una ruta con el menor consumo de combustible posible mientras llegan dentro de un cierto intervalo de tiempo.

3.1.1. Concepto de ruta meteorológica

El *ruteo meteorológico* es una disciplina que consiste en diseñar una ruta óptima para un viaje marítimo, teniendo en cuenta las condiciones climáticas previstas y las características operativas del barco. El objetivo principal es minimizar el costo operativo, considerando la seguridad del barco, la tripulación y la carga, así como las emisiones atmosféricas en ciertas áreas de control. Además del consumo de combustible, el costo operativo incluye otros factores como los salarios de la tripulación y los costos de capital. El costo también depende del tiempo de llegada, que puede estar sujeto a restricciones o penalizaciones.

El *ruteo meteorológico* requiere de un sistema que integre varios elementos, como el modelo de consumo de combustible, los algoritmos de optimización, las restricciones, los pronósticos meteorológicos y las respuestas del barco. El modelo de consumo de combustible debe estimar el gasto energético en función de la resistencia al avance del barco, que depende de factores como el viento, las olas, las corrientes, el ensuciamiento y el mantenimiento del motor. Los algoritmos de optimización deben buscar la ruta que minimice el costo operativo, respetando las restricciones impuestas por la seguridad, el tiempo de llegada y otras consideraciones. Las restricciones pueden incluir aspectos como la navegabilidad del barco, los movimientos y aceleraciones máximas admisibles, la estabilidad, el riesgo de fenómenos como el balanceo paramétrico o el "surf-riding", y la evitación de zonas con hielo, piratería o regulaciones ambientales estrictas. Los pronósticos meteorológicos deben proporcionar información fiable y actualizada sobre las condiciones climáticas esperadas en la ruta, incluyendo la dirección y velocidad del viento, la altura y dirección de las olas, la intensidad y dirección de las corrientes, y la temperatura del aire y del agua. Las respuestas del barco deben modelar el comportamiento dinámico del barco ante las condiciones climáticas, considerando aspectos como las funciones de transferencia (modelos matemáticos que relacionan la entrada y la salida del barco), los espectros de oleaje, los coeficientes hidrodinámicos (miden la resistencia y propulsión del barco) y las propiedades físicas y estructurales del barco.

Cabe señalar que los petroleros y los graneleros pueden tener más flexibilidad en la hora prevista de llegada que los portacontenedores y los transbordadores, que deben cumplir con horarios más estrictos. Por otro lado, un buque de pasajeros debe priorizar la comodidad durante la navegación en su planificación de ruta. Por lo que el tipo de barco y su función es importante a la hora de tener en cuenta la optimización.

Para encontrar la mejor ruta de un buque en un tiempo aceptable, se aplica un algoritmo de búsqueda, también conocido como algoritmo de optimización. El algoritmo tiene en cuenta cuatro áreas principales que afectan al espacio de soluciones, como se muestra en la Fig. 3.1. Estas áreas son: restricciones, factores de combustible, datos genéricos y datos del caso. Las restricciones son los factores que restringen el espacio de soluciones y pueden hacer que algunas soluciones no sean factibles. Los factores de combustible son los factores ambientales que influyen en el consumo de combustible. Los datos genéricos son los datos que no dependen del buque, y los datos del caso son las propiedades del buque y los datos propios del viaje.

3.1.2. Modelo de consumo de fuel

El consumo de combustible de los buques es un factor clave para la eficiencia energética, la reducción de emisiones y el ahorro de costes en el transporte marítimo. Por ello, es importante estimar el consumo

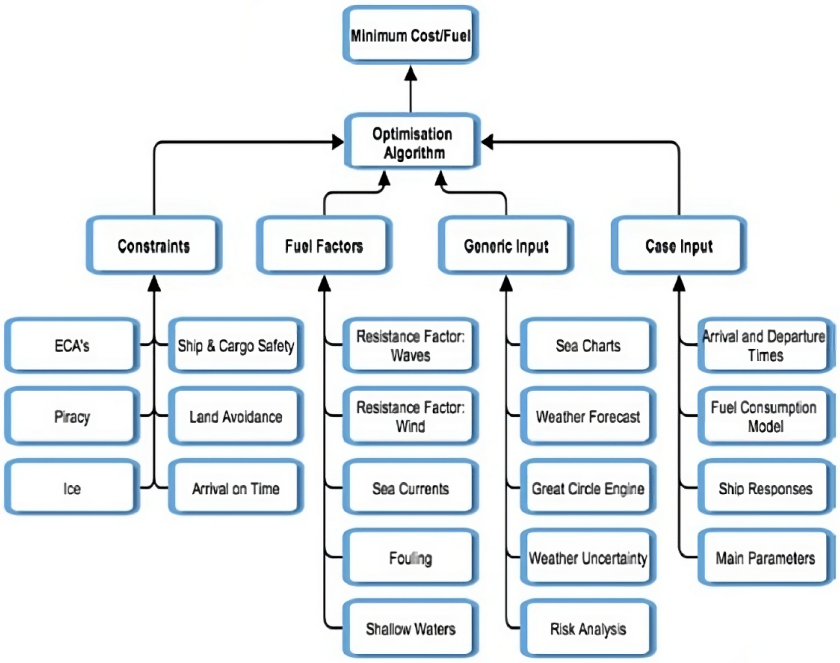

Figura 3.1: Estructura general de un sistema de rutas meteorológicas. Fuente [1].

de combustible de forma precisa y fiable, teniendo en cuenta los factores que lo afectan, como la resistencia al avance, la potencia propulsora, la velocidad del buque, las condiciones ambientales y el estado del casco y la hélice[1].

Existen diferentes modelos para estimar el consumo de combustible de los buques, que pueden basarse en métodos teóricos, empíricos o híbridos. Algunos ejemplos de modelos teóricos son los basados en la ecuación de Holtrop y Mennen [54], que calcula la resistencia al avance en aguas tranquilas y con oleaje, o los basados en la dinámica de fluidos computacional (CFD), que resuelve las ecuaciones de Navier-Stokes para obtener la resistencia al avance y la potencia propulsora. Estos modelos requieren información detallada sobre las características del buque y la geometría del casco, y pueden ser muy precisos pero también muy costosos computacionalmente. El primer paso en la estimación del costo de combustible es obtener la resistencia en aguas tranquilas basada en las características de la embarcación. Durante las condiciones de ruta de la embarcación en el mar, también se agrega resistencia debido a cargas de viento y oleaje. La suma de la resistencia en aguas tranquilas y la resistencia agregada se llama resistencia total. La resistencia de un barco debe ser compensada por la fuerza de empuje proporcionada por las hélices, que son impulsadas por el motor marino con ciertas RPM (revoluciones por minuto) de potencia motor que determinan el consumo de combustible bajo diferentes condiciones de trabajo[55].

La potencia requerida para empujar un barco para superar su resistencia total se llama potencia efectiva, que proviene de la potencia del eje transferida desde el motor marino. La reducción de la potencia del eje a la potencia efectiva está determinada por la eficiencia del casco, la eficiencia de la hélice y la eficiencia del eje, mientras que la potencia del eje se genera a partir del motor marino con una reducción de potencia llamada eficiencia del motor.

Los modelos empíricos se basan en datos históricos o experimentales, como los obtenidos de pruebas en tanque o de mediciones a bordo. Estos modelos pueden utilizar técnicas estadísticas o de aprendizaje automático para ajustar una función que relacione el consumo de combustible con las variables de entrada, como la velocidad del buque, el estado del mar o el régimen del motor. Estos modelos pueden ser muy rápidos y fáciles de implementar, pero también pueden tener limitaciones en cuanto a la calidad y cantidad de los datos disponibles, y pueden perder precisión cuando se extrapolan fuera del rango de los datos.

Los modelos híbridos combinan elementos teóricos y empíricos, tratando de aprovechar las ventajas de ambos enfoques. Por ejemplo, se puede utilizar un modelo teórico para calcular la resistencia al avance y la potencia propulsora, y luego aplicar un factor de corrección empírico para tener en cuenta el efecto del ensuciamiento del casco o la hélice. O se puede utilizar un modelo empírico para estimar el consumo de combustible a partir de datos medidos a bordo, y luego aplicar un factor de corrección teórico para tener en cuenta el efecto del viento o las corrientes.

Uno de los factores ambientales que influye en el consumo de combustible es la resistencia debida al viento. Esta resistencia depende de la forma y el tamaño del buque, así como de la dirección y velocidad del viento. Algunos modelos para estimar la resistencia al viento son los propuestos por Holtrop y Mennen[54] , Fujiwara et al.[56] o Kristensen y Lützen[57]. Estos modelos se basan en ensayos en túnel de viento o en coeficientes empíricos obtenidos a partir de una gran cantidad de datos. Un aspecto que puede aumentar la complejidad de los cálculos es el apilamiento de contenedores en los buques portacontenedores, que afecta a la resistencia al aire y por tanto al consumo de combustible .

Otro factor ambiental que afecta al consumo de combustible es la corriente marina. La corriente puede tener el mismo sentido o el contrario al de la navegación del buque, lo que implica un aumento o una disminución de la velocidad sobre el suelo (SOG) y por tanto del tiempo total de navegación. Esto se traduce en un mayor o menor consumo de combustible en comparación con una situación sin corriente. Un ejemplo es la corriente Kuroshio, que afecta a la velocidad media hasta dos nudos[58].

Para resolver este problema se pueden utilizar diferentes algoritmos de optimización, que se presentarán en 3.3. Estos algoritmos requieren una función objetivo que evalúe el consumo de combustible para cada solución candidata, y por tanto dependen del modelo de consumo de combustible utilizado. Por ello, es importante seleccionar el modelo y el algoritmo más adecuados para cada caso, teniendo en cuenta los objetivos, las restricciones y los datos disponibles [59].

3.1.3. Restricciones

Las restricciones a una ruta, como la tierra, las zonas de tormentas más severas con limitaciones de velocidad, son fácilmente cuantificables, mientras que otras, como los grandes movimientos del buque debido al clima y la dirección de navegación, requieren análisis más complejos. Las predicciones de la respuesta del buque son esenciales para extender el enrutamiento desde la evitación de tormentas hasta los criterios limitantes basados en los movimientos y las aceleraciones[60].

Las respuestas se predicen utilizando las funciones de transferencia, Operadores de Amplitud de Respuesta (RAOs), que se utilizan para determinar principalmente los movimientos de un buque bajo diferentes condiciones de oleaje. El entorno de oleaje encontrado para las respuestas del buque puede ayudar a juzgar si se deben tomar medidas adicionales para mejorar la estabilidad y la seguridad de la carga del buque, etc. Los RAOs se pueden calcular mediante una variedad de métodos, pero todos ellos requieren información o modelado específico del buque. Incluso con unas funciones de transferencia modeladas, la difícil tarea de predecir las respuestas perjudiciales y no lineales de los buques permanece.

Además, las restricciones típicas también pueden provenir de las vibraciones de alta frecuencia del buque, que a menudo causan problemas de confort e integridad estructural. Estas vibraciones de alta frecuencia suelen estar inducidas por cargas de golpeo en proa y popa, vibraciones resonantes del casco del buque, etc., cuyos efectos se pueden evitar o mediante la reducción voluntaria de la velocidad o el cambio del rumbo de una manera bastante rápida. Otro tema importante en la industria marítima es la evitación del balance paramétrico [61], que debe ser tratado como restricciones en las operaciones del buque. Otros factores que pueden o no ser incluidos como restricciones al enrutamiento son las ECAs (Áreas Controladas por Emisiones), las aguas con piratería y el hielo, que en casos normales pueden ser tomados en cuenta por el sistema de enrutamiento como objetivos relacionados con el control de

emisiones atmosféricas, la seguridad y la protección. Las restricciones más recientes provienen de las directrices revisadas por la OMI 1228 [62] para evitar condiciones peligrosas en tiempo climatológico adverso.

3.1.4. Pronóstico de la climatología

La calidad y disponibilidad de los datos meteorológicos procedentes de distintas fuentes es un factor clave para un sistema de rutas. Para diseñar una ruta óptima, se puede necesitar información sobre el viento, el oleaje, la corriente, la temperatura, etc. Un sistema de enrutamiento con datos marítimos fiables puede reducir significativamente las condiciones marítimas adversas a las que se enfrenta un buque durante su travesía [63][64].

Los institutos meteorológicos desempeñan un papel vital al proporcionar pronósticos del tiempo confiables para propósitos de enrutamiento. Estos institutos ejecutan modelos climatológicos globales y locales que toman datos de diversas fuentes, como satélites, boyas, estaciones de medición meteorológica y observaciones a bordo de los buques.

Sin embargo, los pronósticos del tiempo disponibles en la actualidad tienen limitaciones en cuanto al período de tiempo de sus predicciones. Los pronósticos que se extienden más allá de 14-16 días pueden no incluir los parámetros necesarios para fines de enrutamiento, como solo proporcionar distribuciones de presión atmosférica, que no son aplicables para propósitos de enrutamiento.

Existen dos tipos de pronósticos del tiempo utilizados para la optimización del enrutamiento: pronósticos determinísticos y pronósticos del tiempo en conjunto. El pronóstico determinístico se calcula a partir del modelo climatológico utilizando condiciones iniciales de los estados meteorológicos actuales, sin considerar las incertidumbres asociadas con estas mediciones. Puede predecir información meteorológica confiable para uno o dos días, pero el pronóstico a largo plazo (hasta 2 semanas) se vuelve incierto debido a fuentes como condiciones iniciales inciertas, condiciones límite inciertas e imperfecciones del modelo climatológico en sí [1].

Para abordar las incertidumbres asociadas con el modelo determinístico, los sistemas de enrutamiento modernos utilizan pronósticos del tiempo en conjunto. Este método agrega pequeñas perturbaciones al análisis de operación del modelo climatológico, lo que resulta en una serie de pronósticos del tiempo con igual probabilidad de ocurrencia [65]. Sin embargo, algunos estudios sugieren que la optimización del enrutamiento utilizando pronósticos del tiempo en conjunto no produjo mejores resultados que utilizar el pronóstico determinístico estándar [66].

Otro enfoque para obtener pronósticos del tiempo confiables es asimilar datos meteorológicos de diversas fuentes y compensar las incertidumbres a través del análisis de sensibilidad. Se espera que este método genere información meteorológica más confiable y de alta resolución para la planificación del enrutamiento de los buques. Sin embargo, la falta de pronósticos confiables puede requerir que parte del viaje se calcule utilizando datos meteorológicos estadísticos [67].

Una previsión meteorológica sólida es la variable más importante para obtener buenos resultados de ruta, por lo que tener acceso a las mejores disponibles es crucial. Además de la precisión, la resolución varía significativamente entre los distintos proveedores.

3.2. El problema de optimización: Problema Multiobjetivo

La formulación de un método de optimización depende de la elección del algoritmo de optimización. La estructura exacta puede variar, pero la mayoría de las veces se basa en el algoritmo que muestrea

soluciones donde se calcula el consumo específico de combustible y se penaliza o rechaza en función de la violación de restricciones. En la Fig. 3.2 muestra un ejemplo de dicha estructura implementada por Larsson y Simonsen [68]. En dicho artículo, se propone un enfoque basado en parámetros meteorológicos y de barco para optimizar las rutas marítimas. Se utiliza un algoritmo de optimización que tiene en cuenta el pronóstico del tiempo, los parámetros del barco, la hora y el puerto de salida y llegada para diseñar una ruta con el menor costo operativo. El ETA (Tiempo Estimado de Llegada) no se considera como una restricción, sino que se penaliza si el barco no puede llegar a tiempo. Debido a la falta de fiabilidad del pronóstico del tiempo a largo plazo, se actualiza la ruta optimizada cada 12 o 24 horas en función de la nueva información del pronóstico del tiempo.

Por tanto, el algoritmo de optimización se presenta como el componente fundamental del sistema de rutas meteorológicas de alta calidad. Se requiere una revisión detallada de los distintos algoritmos y enfoques implementados en los trabajos de investigación que se han llevado a cabo del tema, con el fin de analizar tanto sus beneficios como sus limitaciones. Estos se analizarán en el punto 3.3.

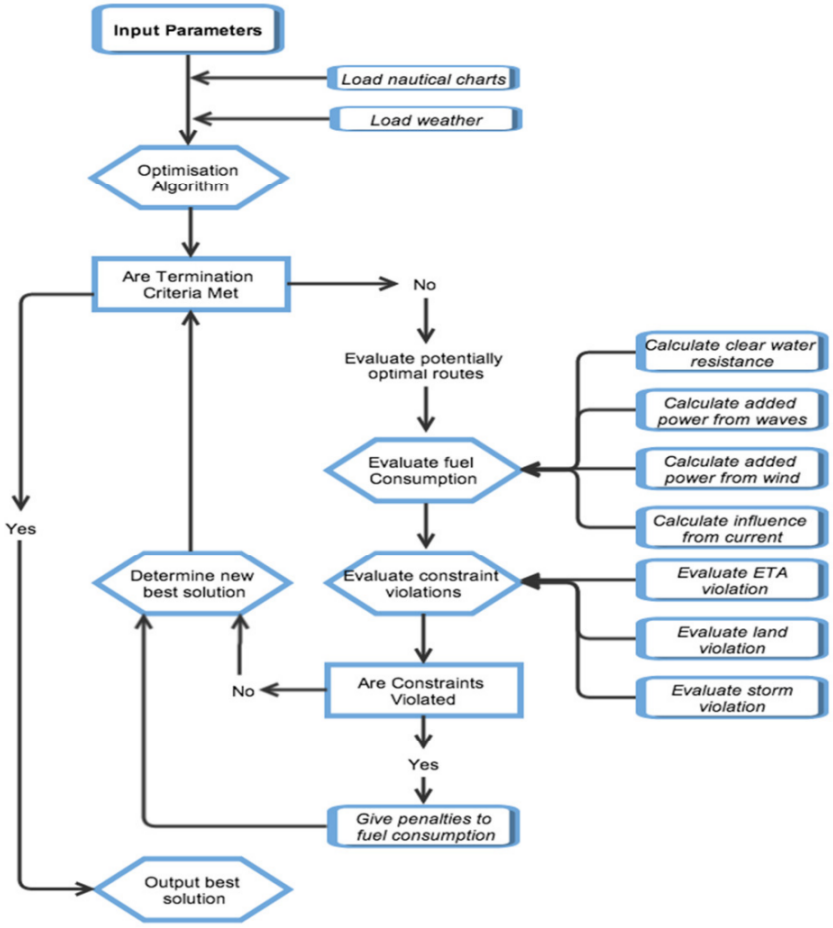

Figura 3.2: Posible implementación de un algoritmo de optimización de rutas marítimas con el objetivo de minimizar el fuel. Fuente [1].

Observando la Fig. 3.3, el núcleo del problema consiste en un conjunto de variables que determinan la ruta, las cuales dependen de dos elementos previos: los criterios de optimización de la ruta y las restricciones del área de navegación, representados en color naranja y azul, respectivamente.

A partir de la entrada de estos elementos, las variables que definen la ruta establecen un trayecto marítimo en verde.

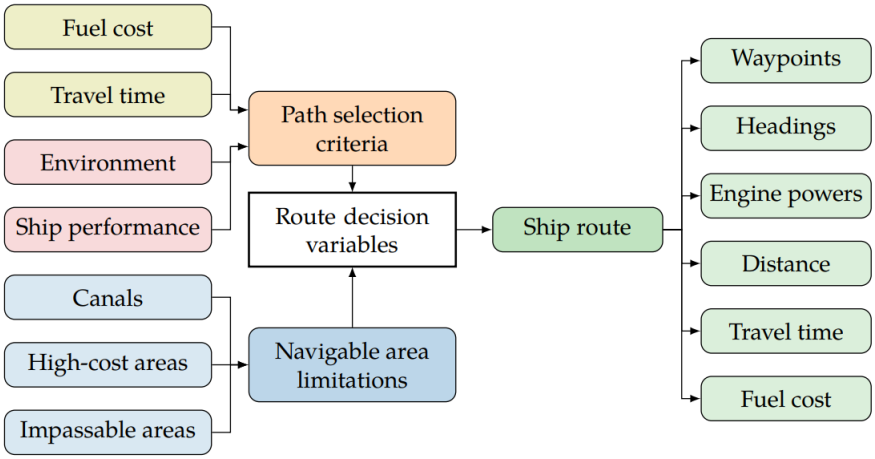

Figura 3.3: Estructura general del Problema de Optimización compuesto de un conjunto de variables que definen la ruta, las cuales están condicionadas por dos factores previos: los criterios de selección de la ruta (naranja) y las limitaciones del área navegable (azul). Los primeros factores incluyen el objetivo de optimización (amarillo) y la evaluación del rendimiento del barco (rojo). El segundo factor, las limitaciones del área navegable, establece las restricciones y los costes para el área de navegación. Una solución al problema es una ruta marítima (verde), que tiene varias características. Fuente: [2]

3.2.1. Ruta Marítima

Una ruta marítima se compone de una serie de puntos(waypoints), desde la salida hasta la llegada. Cada punto tiene unas coordenadas de longitud y latitud en grados. En cada punto, salvo en el final, la ruta indica la dirección y la potencia del barco para alcanzar el siguiente punto. La dirección es el ángulo del arco que forma la proa del barco con el norte.

3.2.1.1. Criterios de selección de rutas

Los criterios de selección de la ruta (naranja) del problema de ruta marítima multiobjetivo (MOSWR) incluyen dos componentes principales: (1) objetivos y (2) evaluación del rendimiento del barco. Los primeros son la minimización del coste de combustible y del tiempo de viaje (amarillo). Por otro lado, la evaluación del rendimiento de la embarcación se ve afectada por las condiciones ambientales y el modelo de rendimiento de la embarcación (rojo).

Objetivos

En el ámbito del transporte marítimo comercial, la navegación de las embarcaciones se basa en un solo criterio o en una combinación de varios criterios. En este Trabajo Fin de Grado, consideramos el problema de minimizar simultáneamente el tiempo de viaje y el costo total de combustible. Estos objetivos son contradictorios, ya que el costo total de combustible tiene una relación negativa con el tiempo de viaje, es decir, minimizar el consumo de combustible conlleva un aumento en la duración del viaje. Por lo tanto, la optimización simultánea de estos dos objetivos requiere una optimización multiobjetivo.

Además de los objetivos, otros elementos que afectan las decisiones de ruta son las previsiones meteorológicas para el área de navegación futura y el rendimiento de la embarcación. Las condiciones ambientales que afectan las decisiones de ruta son el viento, las olas y las corrientes oceánicas. Tanto el viento como las olas afectan principalmente la velocidad de la embarcación, mientras que las corrientes

oceánicas afectan tanto la velocidad como la dirección de la embarcación. Estas condiciones ambientales se consideran en el problema de selección de rutas marítimas.

Por lo tanto, optimizar estos dos objetivos simultáneamente requiere una optimización multiobjetivo. Es poco probable que exista una única solución óptima para un problema multiobjetivo; más bien es un conjunto de soluciones equivalentes. Por lo que se busca encontrar un conjunto de rutas Pareto óptimas desde cualquier salida a cualquier destino. Una ruta Pareto óptima implica una ruta factible con valores mínimos de los objetivos, tal que mejorar un objetivo solo es posible empeorando otro. Se entrará en detalle del concepto de optimalidad de Pareto en la Sección 3.4.9.

Condiciones Ambientales

Las condiciones ambientales que influyen en las decisiones para una ruta mínima de combustible y tiempo son el viento, las olas y las corrientes oceánicas. James(1957)[69] afirma que una combinación de viento y olas genera una resistencia adicional al barco, lo que resulta en una pérdida de velocidad. Por otro lado, las corrientes oceánicas afectan tanto a la velocidad como al rumbo del barco. Las olas (y el viento) afectan principalmente a la velocidad del buque, mientras que las corrientes oceánicas afectan tanto a la velocidad del buque como a su rumbo sobre tierra.

Con una previsión precisa de las condiciones ambientales, se puede realizar una ruta marítima en la que el tiempo estimado de viaje sea más fiable y se minimice. Estimaciones precisas del tiempo de viaje implican menos costes por espera y retraso en el destino, así como una reducción del coste del viaje.

Rendimiento del buque

Las condiciones ambientales también influyen en el rendimiento de la embarcación. Un modelo de rendimiento de la embarcación predice el comportamiento de la embarcación debido a estos efectos y lo traduce en movimientos de la embarcación. Este modelo aproxima el consumo de combustible y la velocidad de la embarcación en diferentes condiciones ambientales. Para que el problema de enrutamiento de la embarcación sea genérico para diferentes tipos de embarcaciones y condiciones ambientales, se debe prestar atención al modelado del rendimiento de la embarcación.

La exactitud de los resultados de la optimización depende en gran medida de la exactitud del modelo de rendimiento del buque bajo diferentes condiciones ambientales y navegación. La precisión de la resistencia adicional debida a las olas y el viento es importante para obtener resultados precisos de optimización del consumo de combustible.

En este Trabajo Fin de Grado se simplificará el modelo de rendimiento del buque y velocidad en las diferentes condiciones, mediante una aproximación mediante un modelo de aprendizaje automático (ML) basándose en datos históricos del sistema de identificación automática (AIS) y los datos meteorológicos marítimos propuesto en M. Abebe et al., 2020 [10] para predecir la velocidad del buque sobre el suelo (SOG) , y cuyo fundamento teórico de los modelos se puede ver en la Sección 3.5 y su modelización, resultados y discusión en el Capítulo 6.

3.2.1.2. Limitaciones de las zonas navegables

Las limitaciones del área navegable (azul) que afectan a una ruta marítima se clasifican en tres componentes: (1) áreas intransitables, (2) áreas de alto coste y (3) canales marítimos principales.

Áreas intransitables

Un barco puede navegar libremente dentro del área navegable, un espacio continuo predefinido que contiene áreas de alto coste y excluye áreas intransitables. Las áreas intransitables son obstáculos terrestres, hielo marino y zonas restringidas, como las líneas costeras y los círculos (ant)árticos.

Áreas de alto coste

Dentro de las áreas de alto coste, se incurre en una tasa adicional de coste operativo, lo que se traduce en costos de combustible adicionales. Por ejemplo, se trata de áreas con riesgos de seguridad, como zonas de piratería o climatología adversa estacional, y áreas con altos costes operativos como las Áreas de Control de Emisiones (ECA).

Las ECAs, como el Mar Báltico, el Mar del Norte y las costas de los Estados Unidos, tienen controles más estrictos para minimizar las emisiones contaminantes de los barcos. Las compañías navieras pueden cumplir con las regulaciones de las ECAs cambiando de combustible o utilizando sistemas de depuración. El cambio de combustible es una opción de cumplimiento directo para los barcos que operan tanto dentro como fuera de las ECAs. El combustible de ultra bajo contenido de azufre (ULSFO) se consume dentro de las ECAs, mientras que se utiliza combustible de muy bajo contenido de azufre (VLSFO) en otras áreas. El ULSFO es más costoso que el VLSFO, lo que puede influir en las decisiones de ruta y velocidad de los barcos [2].

Los costos de combustible dependen en gran medida de la velocidad del motor de la embarcación, ya que el consumo de combustible es aproximadamente proporcional al cubo de la velocidad del barco. Por lo tanto, las compañías navieras que operan tanto dentro como fuera de las ECAs enfrentan diferentes decisiones de velocidad en cada área. Las regulaciones de las ECAs también pueden afectar la trayectoria de la ruta de la embarcación, ya que evitar las ECAs puede resultar en menores costos de combustible.

Además de las ECAs, se introduce una penalización opcional por navegar en aguas poco profundas, lo que implica limitaciones de calado para los barcos. Las rutas preferiblemente evitan estas áreas, pero aún pueden atravesarlas a un costo más alto.

En este Trabajo de Fin de Grado nos centraremos en penalizar las zonas con climatología adversa. Por completar, señalamos que se podrían introducir otras áreas de alto coste, lo que se podría llevar para trabajo de investigación futuro.

Canales de Navegación

Los canales marítimos principales son pasos estrechos que conectan dos masas de agua y permiten a los barcos acortar la distancia entre dos puertos. Algunos ejemplos de canales marítimos principales son el Canal de Panamá, el Canal de Suez y el Canal de Kiel. Estos canales suelen cobrar una tarifa por el paso de los barcos, que depende del tamaño, el tipo y la carga del barco. Además, los barcos pueden enfrentarse a tiempos de espera o restricciones de velocidad al atravesar estos canales. Por lo tanto, los canales marítimos principales pueden ofrecer una opción de ruta más rápida pero más cara, o una opción de ruta más lenta pero más barata. La decisión de pasar o evitar estos canales depende del equilibrio entre el tiempo de viaje y el coste de combustible [2].

Al proporcionar una variedad de soluciones al problema de selección de rutas marítimas, se pueden incluir rutas que pasen por un canal y rutas alternativas que lo eviten, para apoyar la toma de decisiones consciente del usuario final en términos de tiempo de viaje y costos.

3.2.1.3. Formulación Matemática

En esta Sección se presenta el problema MOSWR como un problema de optimización parcialmente continuo. Para ello, primero definimos la región de navegación donde se representa una ruta. La región de navegación incluye áreas intransitables y áreas donde se aplica un coste o penalización adicional, llamadas áreas de alto coste. Luego describimos la estructura de una ruta marítima que debe cumplir con ciertas limitaciones geográficas y del barco. El objetivo en MOSWR es encontrar un conjunto de rutas óptimas en el sentido de Pareto, que satisfagan estas limitaciones. En este Trabajo de Fin de Grado, como ya hemos comentado, la selección de las rutas óptimas se basa en dos valores objetivos: el tiempo de viaje y el coste total de combustible de cada ruta. Estos valores dependen de la distancia de la ruta y la velocidad real del barco, que está determinada por el rendimiento del barco en las condiciones ambientales.

La trayectoria de una ruta marítima es una curva suave a trozos sobre la superficie de un elipsoide terrestre, es decir, una curva suave que cambia de dirección en puntos discretos, como se muestra en la Fig. 3.4, la cuál está definida por una secuencia de $n+1$ puntos de paso que forman n tramos de ruta. Cada segmento i-ésimo contiene las ubicaciones de los puntos de paso inicial y final y la velocidad nominal del barco constante en el tramo. La primera restricción que se impone a una ruta marítima es la de la región de navegación [3].

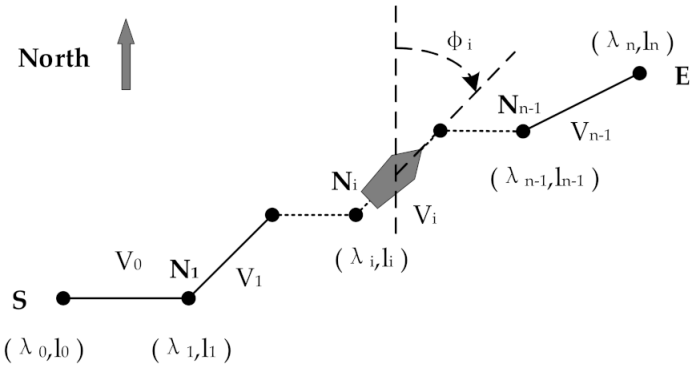

Figura 3.4: Representación esquemática de la ruta y posición actual del buque.
Salida y destino denotados por S y E respectivamente.N_i representa el waypoint inicial del segmento i-ésimo, λ_i y l_i son las coordenadas de latitud y longitud del waypoint i-ésimo N_i respectivamente,V_i es la velocidad en aguas tranquilas del segmento i-ésimo y φ_i es el acimut del segmento i-ésimo. Fuente [3].

Sea la región de navegación Ω_0 proyectada sobre la superficie de un elipsoide terrestre, tal que:

$$\Omega_0 \in \{(\lambda, \varphi) \mid \lambda_{\text{mín}} \leq \lambda \leq \lambda_{\text{máx}}, \varphi_{\text{mín}} \leq \varphi \leq \varphi_{\text{máx}}\} \tag{3.1}$$

donde λ y φ son la latitud y la longitud en grados, respectivamente. Dentro de la región de navegación definimos n_c zonas infranqueables $\Omega_C^i, i \in [n_c]$ que permanecen invariables durante el periodo de navegación. Un barco puede navegar libremente dentro de la región de navegación, pero no puede cruzar las zonas intransitables. Así, podemos expresar la zona navegable Ω_a como

$$\Omega_a = \Omega_0 \backslash \bigcup_{i=0}^{n_c} \Omega_C^i \tag{3.2}$$

Supongamos que la ruta se compone de n segmentos. Cada segmento consta de la ubicación del waypoint de partida, la velocidad establecida en aguas tranquilas y el acimut. Por lo tanto una solución

al problema MOSWR viene definida por el vector de decisión de variables de ruta r, expresado de la forma [3]:

$$\boldsymbol{r} = \{(\lambda_1, l_1, V_1, \varphi_1), \ldots, (\lambda_i, l_i, V_i, \varphi_i), \ldots, (\lambda_n, l_n, V_n, \varphi_n)\} \tag{3.3}$$

siendo respectivamente, λ_i y l_i las coordenadas de latitud y longitud respectivamente del punto de referencia N_i del i-ésimo segmento, V_i es la velocidad establecida en aguas tranquilas del i-ésimo segmento y φ_i es el azimut del i-ésimo segmento.

El par de latitud y longitud de cada waypoint $p \in \mathbf{r}$ están restringidos a estar en Ω_a. No sólo los waypoints deben estar dentro de la zona navegable, sino también toda la trayectoria de la ruta p formada por estos waypoints debe estar dentro de la zona navegable, tal que

$$\mathrm{p} \in \Omega_a \tag{3.4}$$

En otras palabras, una ruta de barco es inviable si existe una intersección de cualquier tramo de ruta $r \in \mathbf{r}$ con cualquier zona infranqueable $\Omega^i_{C'} i \in [n_c]$.

La velocidad nominal del buque V_i en cada tramo constituye la variable de velocidad $\mathbf{v}$. La velocidad deseada constante para cada tramo de la ruta está representada por el vector:

$$\mathbf{v} = (V_1, V_2, \ldots, V_{n-1}, V_n), \quad V_i \in [V_{\text{mín}}, V_{\text{máx}}] \, \forall i \in [n]. \tag{3.5}$$

Sea L la longitud total de la ruta. La longitud de la ruta puede calcularse como la suma de la longitud de cada tramo r_i en r,

$$L = L(\mathbf{r}) = \sum_{i=1}^{n} L_i \tag{3.6}$$

donde L_i es la longitud del i-ésimo segmento y puede calcularse utilizando la distancia de la línea de rumbo (rhumb line) en las Eq. 3.7,3.8 y 3.9.

Tratemos la Tierra como una esfera estándar. La dirección de la línea de rumbo φ para los dos puntos de coordenadas en la proyección de Mercator se puede calcular de la siguiente manera [3]:

$$l_d - l_s = tan\varphi \times \left[\ln \tan\left(\frac{\pi}{4} + \frac{\lambda_d}{2}\right) - \ln \tan\left(\frac{\pi}{4} + \frac{\lambda_s}{2}\right)\right]. \tag{3.7}$$

donde λ_s y l_s son la latitud y la longitud del punto de partida, respectivamente. λ_d y l_d son la latitud y longitud del punto final, respectivamente. Sea L_r la distancia en línea de rumbo entre dos puntos. Se puede calcular de la siguiente manera:

$$L_r = R \times (\lambda_d - \lambda_s) \times \sec\varphi \tag{3.8}$$

Si el barco navega paralelo a la trama (lineas de igual latitud, es decir, cuando la dirección del barco es 90^o o 270^o), siendo $\lambda_s = \lambda_d$, entonces la ecuación de la distancia de la línea de rumbo se convierte en:

$$L_r = R \times (l_d - l_s) \times \cos\lambda_s \tag{3.9}$$

donde R es el radio medio de la Tierra.

El tiempo total de viaje T_{viaje} desde la salida hasta el destino puede sumarse mediante el tiempo t_i empleado en cada segmento y se representa mediante la ecuación:

$$T_{\text{viaje}} = \sum_{i=1}^{n} t_i, \quad t_i = \frac{L_i}{V_a^i} \tag{3.10}$$

donde V_a^i es la velocidad real del segmento i-ésimo, cuya forma de abordar su valor se discutirá en la Sección 3.5.1.

Los múltiples segmentos en los que se compone la ruta hacen que el consumo total de combustible del barco son la suma acumulativa del consumo entre los puntos del segmento.

El consumo total de combustible se puede determinar mediante la siguiente fórmula:

$$f_{\text{fuel}} = \sum_{i=1}^{m-1} (t_i \times FCPH) \tag{3.11}$$

donde f_{fuel} es el consumo total de combustible de la ruta del barco, m es el número total de puntos de referencia (diamante y círculo), t_i es el tiempo de navegación del barco a lo largo del i-ésimo segmento de ruta, y FCPH es el consumo promedio de combustible por hora del barco.

Por lo que siguiendo el modelo empírico propuesto en *WinGD (2016)* [13] y asumiendo una potencia de motor del 70 %, la constante que asumimos como consumo promedio es:

$$\text{FCPH} = 154g/kWh \times 33200kW \tag{3.12}$$

3.3. Trabajos Relacionados

La búsqueda de la navegación de tiempo mínimo y costo mínimo (combustible) ha sido objeto de una extensa investigación durante más de seis décadas. La Fig. 3.5 ilustra el desarrollo de la investigación sobre diferentes métodos de enrutamiento de buques.

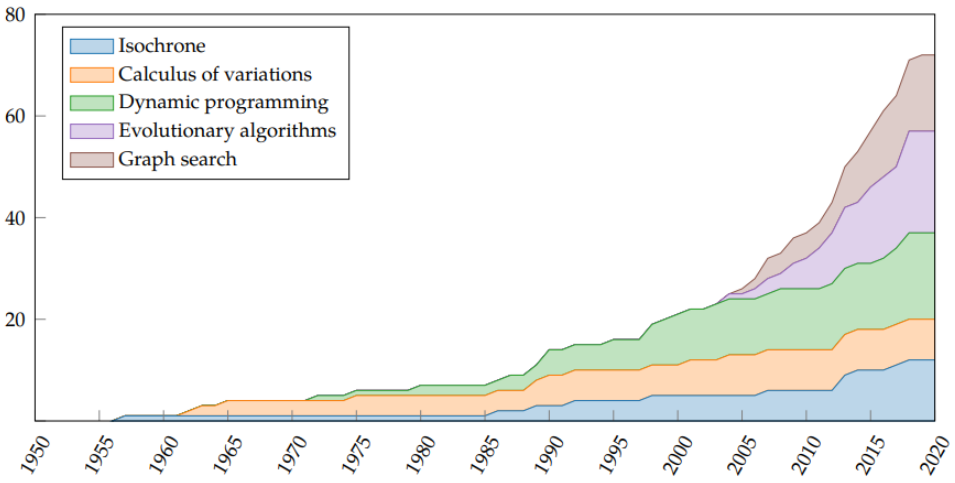

Figura 3.5: Evolución de la investigación sobre cinco métodos generales de ruta marítima. Se recopilan los artículos académicos más relevantes en el periodo 1957-2020. Fuente [2].

Uno de los pioneros fue R.W. James, quién aplicó el método isocrónico (basado en isócronas) para resolver el problema de planificación de rutas de barcos bajo condiciones meteorológicas [69]. Sin embargo, este método presenta el problema de "bucle isocrónico", lo que lo hace inadecuado para el cálculo por computadora. Para solucionar este problema, Hagiwara et al., 1987 [70] propusieron un método isocrónico modificado, y Lin et al., 2013 [71] propusieron un método isocrónico tridimensional modificado. Estos dos

métodos consideran el consumo mínimo de combustible y el tiempo mínimo de navegación como objetivos de optimización. La ruta se optimiza, pero todavía existe un problema de cálculos complejos.

Smierzchalski et al., 2007 [72] utilizaron el método isocrónico para generar la ruta inicial y utilizaron el algoritmo evolutivo para obtener la ruta óptima del barco. Shao et al., 2012 [73] propusieron un algoritmo de planificación hacia adelante, tridimensional y dinámico y planificaron la ruta con la intención de minimizar el consumo de combustible. Sen [74] utilizó el algoritmo de Dijkstra para resolver el problema de planificación de rutas multicriterio de barcos, centrándose en los objetivos de optimización del tiempo de navegación. Mannarini et al., 2016 [75] utilizaron un método de búsqueda en grafo con pesos de arista dependientes del tiempo para optimizar las rutas de barcos. La ruta óptima puede ser más larga en términos de millas navegadas, pero sigue siendo más rápida y segura que la ruta geodésica entre los mismos puntos de partida y llegada.

Con el desarrollo continuo de algoritmos de optimización inteligente y Big Data, estas tecnologías se utilizan gradualmente para resolver el problema de la planificación de rutas de barcos. Por ejemplo, Wang et al., 2018 [3] utilizaron algoritmos genéticos codificados en tiempo real para planificar rutas de barcos con el objetivo de minimizar el tiempo de navegación y el riesgo. Chuang et al. ,2010 [76] aplicaron el método genético difuso considerando el tiempo de transporte, tiempo de atraque de buques portacontenedores y la planificación de las rutas de barcos. Wang et al., 2020 [77] tuvieron en cuenta la maniobrabilidad del barco y aplicaron un algoritmo genético de doble bucle para lograr una planificación de trayectoria dinámica de los barcos. Vlachos [78] utilizó un algoritmo de *Simulated Annealing* para planificar la ruta óptima del barco en función de datos de viento y oleaje previstos. Tsou [79] utilizó el algoritmo de colonia de hormigas (*Ant Colony*) y el algoritmo genético para planificar la ruta de los barcos con un consumo mínimo de combustible. Zhang et al., 2021 [80] propusieron un algoritmo mejorado de colonia de hormigas multiobjetivo al considerar el tiempo de navegación y el riesgo de navegación como objetivos de optimización y realizar la planificación de rutas de barcos. Vettor et al. [81][82] aplicaron un algoritmo evolutivo multiobjetivo para planificar la mejor ruta meteorológica para barcos. He et al., 2019 [83] generaron una ruta optimizada para el barco basada en datos históricos del sistema de identificación automática (AIS). Aunque la longitud de la ruta generada es ligeramente inferior que la trayectoria real del barco; en un entorno complicado, algunas rutas pueden cruzar obstáculos o estar en aguas poco profundas.

La combinación de la optimización por enjambre de partículas y el algoritmo genético (PSO+GA) tiene un buen efecto en la resolución de problemas de optimización de rutas. Abd-El-Wahed et al., 2011 [6] verificaron la superioridad de la combinación de la optimización por enjambre de partículas (PSO, por sus siglas en inglés) y el algoritmo genético (GA, por sus siglas en inglés) en la resolución de problemas de optimización no lineales. Liu et al., 2018 [84] propusieron un método en el que se combinaron GA y PSO para resolver problemas de planificación de rutas en aguas restringidas con un único objetivo de optimización.

3.4. Algoritmos Genéticos

3.4.1. Introducción

Un algoritmo genético (AG, o GA en las siglas en inglés) es una heurística que se basa en la evolución biológica. Utiliza conceptos basados en los genes para modelar soluciones a problemas y luego aplica operaciones parecidas a la selección natural para aproximar la solución más apropiada [85][86][87].

Los algoritmos genéticos (AGs) surgieron a raíz de la labor conjunta de dos investigadores estadounidenses, John Holland en 1975, y fueron presentados en 1989 por David Goldberg como un enfoque para la optimización de búsqueda global. Estos métodos tienen la capacidad de explorar exhaustivamente todo

el espacio de soluciones de un problema, lo que les permite superar posibles óptimos locales y buscar soluciones óptimas a nivel global. Para comprender el concepto de optimización, es importante considerar que la programación matemática se ocupa de resolver procesos que cuentan con diversas soluciones posibles, pero solo una de ellas representa el óptimo global, es decir, la solución que se ajusta mejor a las condiciones del problema en sí. Cualquier otra solución que se asemeje al óptimo global se considera un óptimo local [88].

Los AG son útiles cuando se intenta resolver problemas que a) son computacionalmente difíciles de resolver de forma óptima y b) no tienen un modelo heurístico sencillo para estimar una buena solución [87]. Estos tipos de problemas suelen ser problemas donde hay una gran cantidad de dimensiones que se influyen mutuamente.

3.4.1.1. Proceso de un algoritmo genético

Hay normalmente seis operaciones básicas que un AG utilizará durante una ejecución. Estas son: inicialización de la población, evaluación de la aptitud, selección de los padres, recombinación, mutación y selección de supervivencia [85][86]. Un AG inicializa la población, evalúa la aptitud de cada individuo en esta población y luego repite el bucle hasta que se cumpla alguna condición de terminación. Las condiciones de terminación suelen ser que se encuentre una solución lo suficientemente buena, que se hayan iterado demasiadas generaciones o que la aptitud no haya aumentado en una cantidad determinada de generaciones. Cada bucle consiste en seleccionar los padres que se reproducirán en la población, recombinar estos padres y producir descendencia, a veces mutar la descendencia, evaluar la descendencia y finalmente seleccionar qué individuos de la población serán eliminados. La Fig. 3.6 describe el flujo de trabajo de un AG.

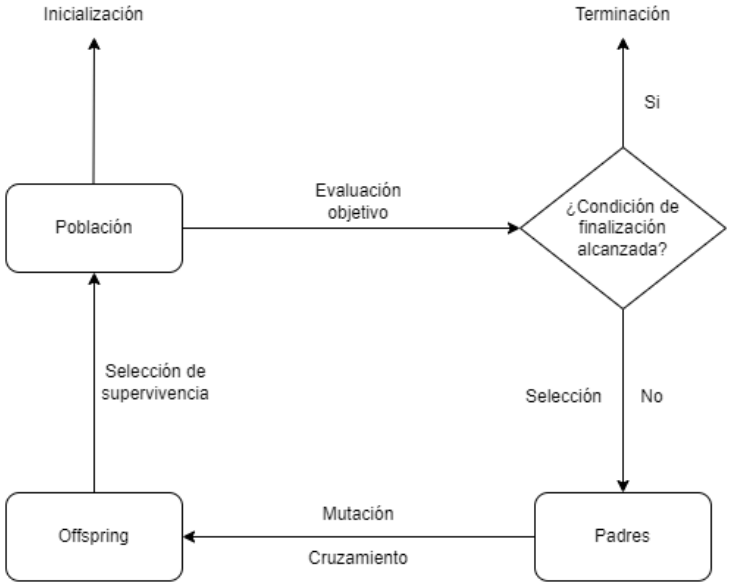

Figura 3.6: Diagrama de flujo de un algoritmo genético.

La terminología para hacer referencia a los procesos de los AGs toma prestados términos del lenguaje biológico. Una solución candidata a un problema se llama *fenotipo*[87] y su representación se llama *individuo o cromosoma* [85] que está formado por un conjunto de *genes*. Es importante reconocer que varios individuos distintos pueden mapear al mismo fenotipo [89], es decir representar a la misma solución. También puede ser posible que los fenotipos representen soluciones inválidas al problema subyacente si por ejemplo se violan algunas restricciones del problema.

3.4.1.2. Ventajas de los algoritmos genéticos frente a heurísticas convencionales

Los algoritmos genéticos presentan ventajas significativas en comparación con las heurísticas convencionales. En primer lugar, son fáciles de implementar y altamente versátiles. La estructura de un algoritmo genético es consistente, lo que simplifica su aplicación en diferentes problemas. La configuración principal que se requiere es la definición de la función de aptitud, que está basada en el problema específico que se desea resolver. Además, las funciones de selección, cruce y mutación ya están predefinidas, lo que facilita su uso, y completamente adaptables al problema que haya que abordar.

Además, los algoritmos genéticos son intrínsecamente paralelos, lo que significa que pueden trabajar y desarrollar múltiples soluciones de manera simultánea. Esta característica es especialmente útil en problemas de búsqueda y optimización, ya que permite explorar el espacio de soluciones en diferentes direcciones al mismo tiempo. Al abordar un problema, los algoritmos genéticos tienen la capacidad de examinar y evaluar múltiples soluciones potenciales, lo que aumenta la eficiencia y la probabilidad de encontrar la mejor solución posible[90].

3.4.2. Población: Inicialización

Al comienzo de la ejecución de un algoritmo genético (AG), es necesario inicializar la población. Una población se refiere al conjunto de todos los individuos evaluados en el AG. Una población tiene un cierto tamaño: un tamaño grande tiene el inconveniente de un mayor tiempo de evaluación, mientras que, un tamaño menor tiene el inconveniente de una falta de diversidad [85][87]. Al inicializar una población, existen dos aspectos importantes a considerar: la diversidad de la población y el direccionamiento hacia soluciones potenciales ya favorables.

La diversidad de la población se refiere a la variabilidad de los individuos que la componen. Cuanto más diversa sea la población, más probable es que no converja tempranamente y se quede atrapada en un máximo local [87]. El otro aspecto consiste en intentar crear una población que esté inicialmente sesgada hacia soluciones favorables. Esto reduce la diversidad de la población, y en ciertos problemas puede no ser posible. Sin embargo, en aquellos casos en los que sea factible, generar una población inicial que apunte a regiones beneficiosas del espacio de búsqueda permitirá reducir la cantidad de trabajo necesaria para encontrar soluciones óptimas.

3.4.3. Función Objetivo

Con el fin de representar la calidad de la solución, se utiliza otro término prestado de la evolución biológica: aptitud (fitness), que describe cuán bien resuelve un individuo el problema subyacente. La aptitud se calcula utilizando una función de aptitud que convencionalmente tiene en cuenta no solo el costo de la solución, sino también el grado de inviabilidad [85].

La aptitud se calcula de acuerdo con alguna función objetivo del problema, donde las soluciones que resuelven mejor el problema suelen obtener una aptitud más alta. Las soluciones que son válidas y se ajustan a las restricciones del problema generalmente tienen una aptitud más alta que aquellas que no lo hacen.

La función de aptitud en problemas simples se puede convertir en una correspondencia directa de lo que se busca en el problema subyacente al valor de aptitud [91]. En otros casos, el problema es más complicado y se necesitan funciones de aptitud más complejas para obtener un buen rendimiento [92].

Para poder utilizar la aptitud, los valores de aptitud deben existir en un continuo donde las soluciones

ligeramente mejores tienen valores ligeramente más altos. De esta manera, un individuo puede tener hijos casi idénticos que realicen pequeños avances en la mejora y se les incentive a seguir mejorando.

Por lo general, evaluar la aptitud de un individuo requiere examinar y aplicar la solución candidata representada al problema subyacente, lo cual puede ser una tarea que consume recursos.

En las ejecuciones de los AGs, es común encontrarse con un problema relacionado con la velocidad de convergencia. En algunos casos, la convergencia ocurre de forma rápida, lo que se conoce como convergencia prematura, donde el algoritmo se enfoca en óptimos locales. Por otro lado, en otros casos el problema es contrario, es decir, se presenta una convergencia lenta del algoritmo [88].

Una posible solución a estos problemas implica realizar transformaciones en la función objetivo. La convergencia prematura suele ocurrir cuando la selección de individuos se realiza de manera proporcional a su función objetivo. En esta situación, puede haber individuos con una adaptación superior al problema en comparación con el resto, lo que lleva a que dominen la población a medida que avanza el algoritmo. Mediante una transformación de la función objetivo, como una comprensión del rango de variación, se busca evitar que estos "superindividuos" dominen la población.

Del mismo modo, el problema de la lenta convergencia del algoritmo puede abordarse mediante un aumento del rango de la función objetivo con la que se busca aumentar la diversidad de la población y acelerar la convergencia hacia soluciones óptimas [88].

Siguiendo un enfoque propuesto por Goldberg y Richardson (1987) [93],la modificación de la función objetivo de cada individuo, de manera que aquellos que estén muy cercanos entre sí tengan una devaluación en su función objetivo, consigue el objetivo de aumentar la diversidad de la población.

Por ejemplo, esta puede verse representada mediante la distancia de Hamming, en la que esta distancia entre los individuos I_t^j e I_t^i sea $d\left(I_t^j, I_t^i\right)$, y denotamos por $K \in \mathcal{R}^+$ un parámetro, podemos definir la siguiente función [88]:

$$h\left(d\left(I_t^j, I_t^i\right)\right) = \begin{cases} K - d\left(I_t^j, I_t^i\right) & \text{si } d\left(I_t^j, I_t^i\right) < K \\ 0 & \text{si } d\left(I_t^j, I_t^i\right) \geq K \end{cases} \tag{3.13}$$

A continuación para cada individuo I_t^j, definimos $\sigma_j^t = \sum_{i \neq j} h\left(d\left(I_t^j, I_t^i\right)\right)$. Este valor se utilizará para devaluar la función objetivo del individuo en cuestión. Es decir, $g^*\left(I_t^j\right) = g\left(I_t^j\right)/\sigma_j^t$. De esta manera, los individuos que están cercanos entre sí verán reducida la probabilidad de ser seleccionados como padres, mientras que se aumentará la probabilidad de los individuos que se encuentran más aislados [88].

3.4.4. Selección

Con el fin de permitir que los individuos se reproduzcan con otros individuos, se emplea un algoritmo que determina qué individuos se aparean, lo que se conoce como selección de padres.

La selección de padres debe considerar el equilibrio entre: 1) permitir que solo los individuos más aptos se reproduzcan para mejorar la calidad de la solución en relación a perder la diversidad de la población y potencialmente limitar el espacio de búsqueda; o 2) distribuir la reproducción entre diferentes individuos para mantener alta la diversidad.

No hay una limitación en la cantidad de padres que se pueden utilizar, y existen investigaciones que indican que el uso de más de dos padres para la reproducción tiene beneficios positivos [94][87][95].

Normalmente, al decidir cuántos padres se reproducirán y cuántos descendientes deben producir, se busca mantener un equilibrio con la cantidad de individuos que son eliminados en cada generación para mantener constante el tamaño de la población.

Hay diferentes tipos de métodos de selección en los AGs [96]:

- *Selección por ruleta:* La probabilidad de elegir un individuo para la reproducción en la siguiente generación es proporcional a su aptitud. Los individuos con mayor aptitud tienen una mayor probabilidad de ser seleccionados.
- *Selección por rango:* La probabilidad de selección no depende directamente de la aptitud, sino del rango de aptitud del individuo dentro de la población. Esto permite considerar las diferencias de aptitud y no requiere conocer los valores exactos de aptitud.
- *Selección por estado estable:* En cada generación se seleccionan unos pocos cromosomas (los mejores en aptitud) para crear una nueva descendencia. Luego, se eliminan algunos cromosomas de baja aptitud y se reemplazan por la nueva descendencia. El resto de la población sobrevive a la nueva generación.
- *Selección por torneo:* Se elige al individuo ganador de cada torneo para realizar el cruce. Consiste en seleccionar aleatoriamente varios individuos y elegir al mejor de ellos para la reproducción.
- *Selección elitista:* Se conserva una pequeña parte de los mejores individuos de la generación anterior en la siguiente generación, sin realizar cambios en ellos. Esto permite mantener características deseables en la población.
- *Selección Boltzmann:* Utiliza una temperatura variable para controlar la tasa de selección según un programa predefinido. La temperatura inicial es alta, lo que significa una baja presión de selección. A medida que la temperatura disminuye gradualmente, aumenta la presión de selección, permitiendo un enfoque más preciso hacia la mejor parte del espacio de búsqueda mientras se mantiene la diversidad adecuada.

3.4.5. Cruzamiento/Recombinación

En los AGs se utilizan dos operadores de variación diferentes: recombinación y mutación. Estos permiten que los individuos cambien y mejoren. La recombinación es la analogía de los AGs a la reproducción, en términos biológicos. Dos o más individuos son seleccionados por el algoritmo de selección de padres y se generan descendientes.

El algoritmo de recombinación determina cómo se crea la descendencia. Existen algunos operadores generales de recombinación que son utilizados por muchos AGs en diversas situaciones. Estos operadores sirven como buenos puntos de referencia para comparar con los operadores de recombinación más específicos del problema.

Los operadores generales son los operadores de cruce de un punto, los operadores de cruce de varios puntos y los operadores de cruce uniforme [85][86][87]:

- *Cruce de un punto:* Se elige aleatoriamente un punto en los cromosomas de ambos padres, al cual se denomina "punto de cruce". Los bits a la derecha de ese punto se intercambian entre los dos cromosomas parentales. Esto resulta en dos descendientes, cada uno llevando información genética de ambos padres.

- *Cruce de dos puntos:* En el cruce de dos puntos, se eligen aleatoriamente dos puntos de cruce de los cromosomas parentales. Los bits entre los dos puntos se intercambian entre los organismos parentales.

  Es equivalente a realizar dos cruces de un punto con diferentes puntos de cruce. Esta estrategia se puede generalizar al cruce de k puntos para cualquier número entero positivo k, eligiendo k puntos de cruce.

- *Cruce uniforme:* En el cruce uniforme, típicamente, cada bit se elige del padre correspondiente con igual probabilidad. En un cruce uniforme, no dividimos el cromosoma en segmentos, sino que tratamos cada gen por separado. En este caso, básicamente lanzamos una moneda para decidir si se incluirá o no en la descendencia cada cromosoma. A veces se utilizan otras proporciones de mezcla, sesgando la moneda hacia un padre para tener más material genético del mismo en el hijo.

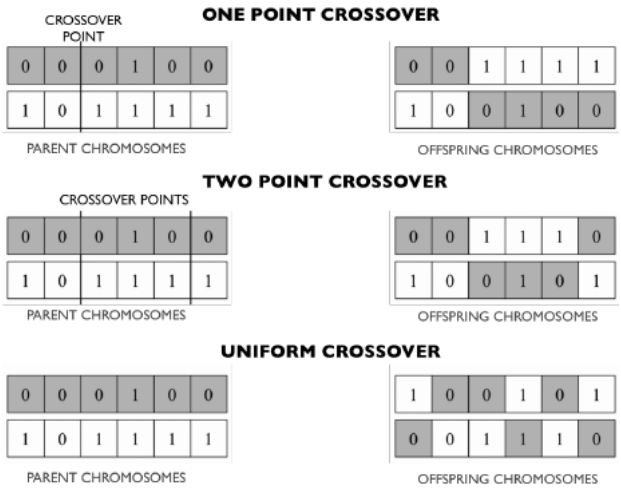

Figura 3.7: Ilustración de ejemplos de métodos de cruce de un punto, dos puntos y uniforme. Fuente: [4]

3.4.6. Mutación

El otro tipo de operador de variación, la mutación, es una operación unaria. Esto significa que la entrada es solo un individuo. La mutación se considera un operador básico que introduce un elemento de aleatoriedad en la vecindad de los individuos de la población. Si bien se reconoce que el operador de recombinación es responsable de explorar el espacio de soluciones potenciales, diversos experimentos realizados por investigadores han demostrado que el operador de mutación adquiere importancia a medida que la población de individuos converge [97].

Las mutaciones se aplican típicamente de manera estadística con una tasa de mutación determinada y realizan cambios pequeños en un individuo. Este cambio también suele ser estocástico y tiene como objetivo crear diversidad dentro de la población sin generar una diferencia tan grande como la recombinación [85][87]. En general, existen operadores de mutación generalizados que se pueden utilizar para casi todos los problemas de AG. Estos tienden a involucrar el intercambio de los valores de los genes en el individuo.

3.4.7. Selección de supervivencia: Reemplazo

La selección de supervivencia es un operador que se encarga de eliminar los individuos sobrantes después de la recombinación y la mutación. Normalmente, la selección de supervivencia intenta estabilizar la población para que no disminuya ni aumente a lo largo de las generaciones. No hay un consenso

claro sobre el mejor tipo de operador de selección de supervivencia, pero en H.Feltl et al., 2004 [98] se argumenta que se obtiene un mejor rendimiento para el problema de asignación eliminando los individuos menos aptos. En PCH Chu, 1997 [85] se argumenta que en lugar de utilizar un valor de aptitud unidimensional, es preferible un enfoque bidimensional con valores de aptitud y no aptitud, ya que la selección de supervivencia funciona mejor si utiliza solo el valor de no aptitud. En El-Ghazali Talbi, 2009[87] se argumenta que a menudo es preferible no eliminar los individuos peores, ya que disminuye la diversidad, y que se debería adoptar un enfoque estadístico.

3.4.8. Condición de finalización

La condición de terminación se verifica al final de cada iteración del algoritmo genético (AG) para determinar si se debe finalizar la ejecución o no. En esta Trabajo de Fin de Grado se utilizó una única condición de terminación diseñada para establecer un límite absoluto para evitar que algunos AGs continúen indefinidamente. La condición de terminación utilizada es de detención absoluta, lo que significa que la ejecución se detiene si la cantidad de generaciones supera un número establecido.

3.4.9. Optimalidad de Pareto: Dominancia

En los problemas multiobjetivo no suelen existir una única solución factible que minimice todas las funciones objetivo a la vez, existe un conjunto de soluciones No-Dominadas que forman la Frontera de Pareto, que consiste en soluciones Pareto-óptimas [2].

Denotando $\mathbf{z} = f(\mathbf{r})$ como el vector de valores objetivos de la solución $\mathbf{r}$. Además, escribamos la ruta $\mathbf{r}_j$ como $\mathbf{j}$ para mayor legibilidad. Una definición subyacente de la optimalidad de Pareto es la noción de dominancia de Pareto. Para cualquier par de individuos $\mathbf{i}, \mathbf{j} \in R$,

$$\mathbf{i} \prec \mathbf{j} (\mathbf{i} \text{ domina a } \mathbf{j}) \Leftrightarrow \mathbf{z}_i < \mathbf{z}_j,$$
$$\mathbf{i} \preceq \mathbf{j} (\mathbf{i} \text{ domina débilmente a } \mathbf{j}) \Leftrightarrow \mathbf{z}_i \leq \mathbf{z}_j.$$

Una solución $\mathbf{r}^* \in R$ se denomina Pareto-óptima si no existe una $\mathbf{r} \in R$ tal que $\mathbf{z} < \mathbf{z}^*$. Si $\mathbf{r}^*$ es Pareto-óptima, $\mathbf{z}^*$ se llama eficiente. El conjunto de todas las soluciones Pareto-óptimas $\mathbf{r}^* \in R$ se denota como R^* y se le llama conjunto de Pareto. Y, el conjunto de todos los puntos eficientes $z^* \in Z$ is Z^*, la frontera de Pareto [2]. Queda ilustrado la frontera de Pareto y las relaciones de dominancia en la Fig. 3.8.

3.4.10. NSGA-II

El Algoritmo Genético de Ordenación No Dominada (NSGA) fue propuesto por primera vez por Srinivas y Deb (1994) [99]. Posteriormente, se presentó NSGA-II por Deb y Goel (2001) [100], mejorando la complejidad computacional e incluyendo una estrategia de preservación de diversidad basada en el uso de compartición en NSGA y en el empleo de crowding en NSGA-II. Además, utiliza un principio elitista al enfatizar soluciones no dominadas. La selección de individuos se basa en su rango de no dominancia y en la distancia de crowding, es decir, la densidad del espacio objetivo alrededor de la solución. La medida crowding es utilizada para seleccionar las soluciones más dispersas entre los individuos del último frente utilizado en la nueva población. Cuanto mayor distancia de crowding de una solución al resto de su frente mejor, ya que la concentración es menor en esa zona.

El algoritmo NSGA-II se basa en la creación de una población descendiente, Q_t, de tamaño N, a partir de la población de padres P_t, también de tamaño N. Estas dos poblaciones se combinan para formar una población total, R_t, de tamaño $2N$. A continuación, se realiza un proceso de ordenamiento no dominado

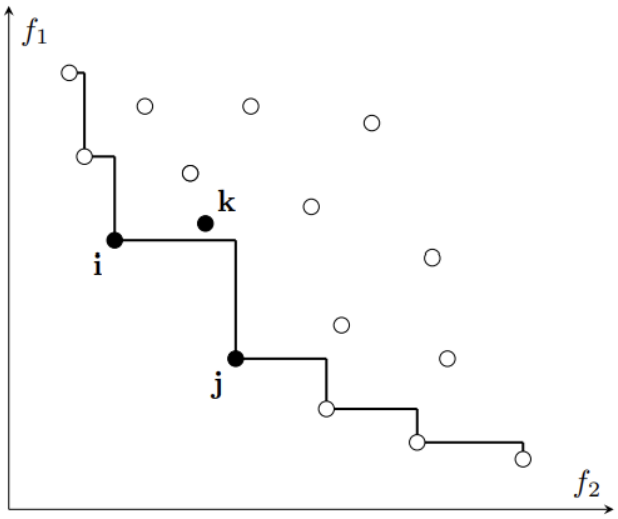

Figura 3.8: Ejemplo frontera de Pareto en el que se minizan dos objetivos. La solución **k** es dominada tanto por las soluciones **i** como por **j**; por lo tanto, su valor no se encuentra en la frontera de Pareto. Los valores objetivo de **i** y **j** se encuentran en la frontera de Pareto, ya que ambas soluciones se dominan débilmente entre sí. Fuente: [2].

en R_t, el cual implica un esfuerzo adicional pero resulta justificado debido a la necesidad de verificar la dominancia de manera global entre los individuos de las poblaciones de padres y descendientes [5].

Una vez finalizado el proceso de ordenamiento no dominado, se procede a generar la nueva población, P_{t+1}, a partir de las configuraciones de los frentes no dominados. La construcción de esta nueva población se inicia con el mejor frente no dominado (F_1) y continúa con los frentes sucesivos (F_2, F_3, etc.). Sin embargo, dado que la población total R_t consta de $2N$ individuos, mientras que la población descendiente solo puede contener N individuos, no todas las configuraciones de los frentes en R_t podrán ser seleccionadas para formar parte de P_{t+1}. En consecuencia, aquellos frentes que no puedan ser acomodados se eliminan por completo [5].

Al llegar al último frente, puede darse el caso de que las soluciones pertenecientes a este excedan el espacio disponible en la población descendiente. Esta situación se representa en la Fig. 3.9. Para abordar esta circunstancia, resulta útil emplear una estrategia que permita seleccionar configuraciones ubicadas en áreas poco pobladas, es decir, alejadas de otras soluciones. De esta manera, se llenan las posiciones restantes en la población descendiente, priorizando la elección de configuraciones de alta calidad en lugar de recurrir a selecciones aleatorias.

Es importante tener en cuenta que este proceso es menos relevante en las primeras etapas generacionales del algoritmo, cuando aún existen numerosos frentes que se mantienen hacia las generaciones siguientes. No obstante, a medida que avanza el proceso, muchas configuraciones pasan a formar parte del primer frente, lo que puede conducir a que dicho frente tenga más de N individuos. Por lo tanto, resulta fundamental que las configuraciones seleccionadas para formar P_{t+1} sean de excelente calidad y sean elegidas de manera que aseguren diversidad dentro del mismo frente de Pareto. De esta forma, cuando la población en su totalidad converge hacia el frente de Pareto óptimo, el algoritmo garantiza que las soluciones estén espaciadas adecuadamente entre sí, promoviendo una representación amplia y diversa de soluciones óptimas.

El algoritmo NSGA-II comienza creando una población inicial de padres, P0, ya sea de forma aleatoria o utilizando una técnica de inicialización. Luego, se ordena esta población en función de los niveles de no dominancia, es decir, se realiza un ordenamiento de los frentes de Pareto (F_1, F_2, etc.). A cada solución se

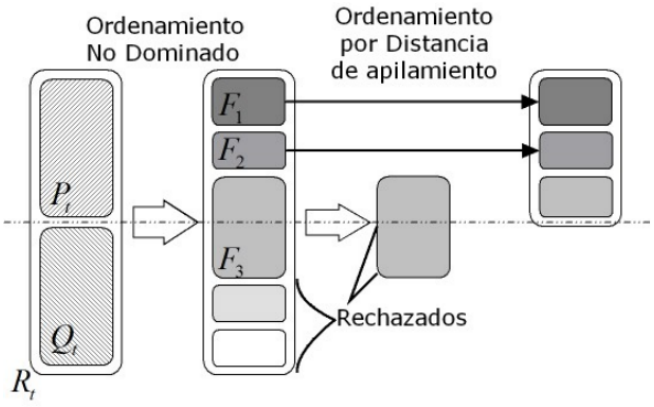

Figura 3.9: Representación esquemática del proceso de selección de individuos del NSGA-II. Fuente: [5]

le asigna una función de aptitud (fitness) basada en su nivel de no dominancia, donde se busca disminuir esta función a medida que avanza el proceso.

La generación de la población descendiente, Q_0, de tamaño N, se realiza mediante la selección por torneo, el cruzamiento y la mutación. Los pasos principales del algoritmo NSGA-II se describen a continuación [5]:

1. Combinar las poblaciones de padres y descendientes para crear $R_t = P_t \cup Q_t$. Realizar el ordenamiento no dominado a R_f e identificar los frentes $F_i, i = 1, 2, \ldots$, etc.
2. Crear una nueva población de padres, $P_{t+1} = \emptyset$, e $i = 1$. Mientras $|P_{i+1}| + |F_i| < N$ hacer $|P_{i+1}| = |P_{i+1}| \cup |F_i|$ e $i = i + 1$.
3. Realizar un ordenamiento por apilamiento ($F_i' < C$) y agregar en P_i las $N - |P_{r+1}|$ soluciones más dispersas usando los valores de distancia de apilamiento asociadas al frente F_i.
4. Generar la población descendiente, Q_{i+1}, a partir de P_{i+1} utilizando la selección por torneo, el cruzamiento y la mutación.

3.4.10.1. Selección por torneo para apilamiento

El operador de selección por torneo para apilamiento ($F_i\prime$ <C) compara dos soluciones y elige un ganador del torneo. Cada solución tiene un rango de no dominancia asociado (r_i) y una distancia de apilamiento (d_i). La distancia de apilamiento d_i es una medida del espacio de búsqueda alrededor de una solución que no está ocupado por otras soluciones en la población. Se utiliza este operador para encontrar la mejor configuración, considerando tanto el rango de no dominancia como la distancia de apilamiento. Si las soluciones se encuentran en el mismo frente de Pareto, se selecciona la que tenga una mayor distancia de apilamiento [5]. La distancia de apilamiento, denotada como $d_{I_j^m}$, según el índice I:

$$d_{I_j^m} = d_{I_j^m} + \frac{f_m^{\left(I_{j+1}^m\right)} - f_m^{\left(I_{j-1}^m\right)}}{f_m^{\text{máx}} - f_m^{\text{mín}}} \tag{3.14}$$

donde $f_m^{\text{máx}}$ y $f_m^{\text{mín}}$ representan el valor máximo y mínimo de la función objetivo m, respectivamente. Además, $f_m^{\left(I_{j+1}^m\right)}$ y $f_m^{\left(I_{j-1}^m\right)}$ son las soluciones vecinas a la configuración j para cada una de las funciones objetivo m.

Las distancias se calculan teniendo en cuenta todas las funciones objetivo y se asigna un valor infinito a las soluciones extremas del frente de Pareto. Estas soluciones extremas son aquellas que tienen el mejor valor en al menos una función objetivo del frente. La distancia resultante es la suma de las distancias en cada una de las direcciones de las funciones objetivo del problema [5].

De esta manera, el algoritmo utiliza operadores genéticos básicos y favorece en el siguiente ciclo generacional a las configuraciones que pertenecen a los mejores frentes y que presentan mayor diversidad, mediante las distancias de apilamiento.

3.4.11. GA+PSO

PSO (Particle Swarm Optimization, o optimización por enjambre de partículas) y AG son algoritmos de optimización heurísticos desarrollados recientemente y ampliamente utilizados en la planificación de rutas.

El algoritmo PSO es un algoritmo de búsqueda aleatoria basado en la cooperación de grupos, motivada por la simulación del comportamiento social. En concreto, cada individuo (agente) utiliza dos tipos importantes de información en un proceso de decisión. La primera es su propia experiencia; es decir, han probado las opciones y saben qué estado ha sido mejor hasta el momento, y saben lo bueno que ha sido. La segunda es la experiencia de otros agentes, es decir, saben cómo les ha ido a otros agentes de su entorno. Es decir, saben qué elecciones de sus vecinos han sido las más positivas hasta el momento y cómo de positivo fue el mejor patrón de elecciones. En el sistema PSO, cada agente toma su decisión en función de sus propias experiencias y de las experiencias de otros agentes.

El sistema tiene inicialmente una población de soluciones aleatorias. A cada solución potencial, llamada partícula (agente), se le da una velocidad aleatoria y se desplaza por el espacio del problema. Los agentes tienen memoria y cada uno de ellos lleva un registro de su mejor anterior (llamada $\mathrm{P_{best}}$) y su aptitud correspondiente. Existe un número de $\mathrm{P_{best}}$ para los respectivos agentes del enjambre y el agente con mayor aptitud se denomina mejor global ($\mathrm{G_{best}}$) del enjambre. Cada partícula se trata como un punto en un espacio n-dimensional. La i-ésima partícula se representa como $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$. La mejor posición anterior de la i-ésima partícula (P_{best}), que proporciona el mejor valor de aptitud, se representa como $P_i = (p_{i1}, p_{i2}, \ldots, p_{in})$. La mejor partícula entre todas las partículas de la población se representa como $P_g = (p_{g1}, p_{g2}, \ldots, p_{gn})$. La velocidad, es decir, la tasa de cambio de posición para la partícula i, se representa como $V_i = (v_{i1}, v_{i2}, \ldots, v_{in})$ [6].

Las partículas se manipulan según las siguientes ecuaciones (los superíndices denotan la iteración):

$$v_i^{k+1} = w \times v_i^k + c_1 \times r_1 \times \left(p_i - x_i^k\right) + c_2 \times r_2 \times \left(p_g - x_i^k\right) \tag{3.15}$$

$$x_i^{k+1} = x_i^k + v_i^{k+1} \tag{3.16}$$

donde $i = 1, 2, \ldots, N$, y N es el tamaño de la población; w es el peso de inercia; c_1 y c_2 son dos constantes positivas, llamadas parámetros cognitivos y sociales, respectivamente; r_1 y r_2 son números aleatorios distribuidos uniformemente en el rango $[0, 1]$. La Eq.3.15 se utiliza para determinar la nueva velocidad v_i^{k+1} de la i-ésima partícula en cada iteración, mientras que la Eq.3.16 proporciona la nueva posición de la i-ésima partícula x_i^{k+1}, sumando su nueva velocidad v_i^{k+1} a su posición actual x_i^k. La Fig.3.10 muestra la descripción de las actualizaciones de velocidad y posición de una partícula en un espacio de parámetros bidimensional. La Fig.3.11 muestra el pseudocódigo del algoritmo general de PSO.

PSO y GA son muy similares en sus características inherentes de paralelismo, mientras que los experimentos muestran que tienen sus ventajas específicas a la hora de resolver problemas diferentes. Con

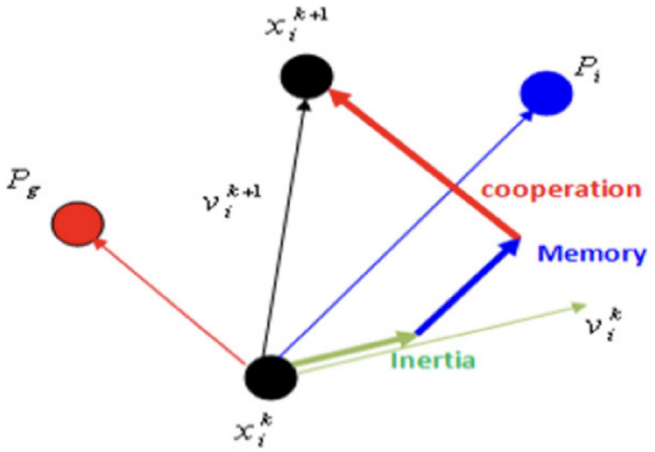

Figura 3.10: Descripción de las actualizaciones de velocidad y posición en la optimización de enjambre de partículas para un espacio de parámetros bidimensional. Fuente: [6].

Randomly initialize positions and velocities of all particles.
Do:
Set P_{best} and G_{best}.
Calculate particle velocity according to equation (2).
Update particle position according to equation (3).
Evaluate the objective function value (fitness value).
while a satisfactory solution has been found

Figura 3.11: El pseudocódigo del algoritmo PSO general. Fuente: [6].

la combinación de ambos, el planteamiento es obtener las excelentes características de ambos integrando los dos algoritmos. Por lo que en este Trabajo de Fin de Grado, se estudiará una combinación GA+PSO multicriterio para resolver el problema de planificación y optimización de rutas marítimas. El algoritmo propuesto seguirá el flujo propuesto en Zhao, W. et al., 2021 [14] que combina principalmente la operación cooperativa de partículas asociada con PSO, la operación de cruce, la operación de mutación y la operación de selección de élite multigrupo en GA y mejora la distribución del conjunto de soluciones de Pareto. El diagrama de flujo puede apreciarse en la Fig. 4.5.

3.5. SOG: Modelos de Machine Learning

3.5.1. Velocidad sobre el suelo (SOG) afectada por las corrientes oceánicas

La velocidad nominal del buque suele reducirse como consecuencia de la resistencia añadida debida al viento y las olas. Esta reducción involuntaria de la velocidad depende de múltiples factores, como los distintos tipos de condiciones ambientales, la hidrodinámica del buque y otras características del buque.

Además de una reducción involuntaria de la velocidad debida al viento y las olas, la velocidad real del buque puede verse afectada por las corrientes oceánicas tanto en su magnitud como en su dirección.

De tal manera que los efectos de estas olas se manifiestan principalmente de dos maneras: en primer lugar, la pérdida natural de velocidad debida a la mayor resistencia del viento y las olas y a la menor la eficacia de la hélice con mal tiempo, y en segundo lugar, la reducción voluntaria de la velocidad, que es la reducción deliberada de la velocidad por parte del capitán del buque para reducir los violentos efectos

del balanceo, el cabeceo y el oleaje,y por tanto, garantizar la seguridad del buque.

En general, hay tres formas de estudiar la reducción natural de la velocidad de un buque con viento y olas. El primero es el método teórico, según el método de balance de energía del sistema agua-casco-aire. Este método utiliza una fórmula para calcular el aumento de la resistencia, con el fin de calcular la disminución de la velocidad a potencia de motor constante. El segundo es el método de prueba, que utiliza experimentos de simulación en tanques y túneles de viento para medir los factores relevantes para determinar el valor de pérdida. El tercero es el método de la fórmula empírica basado en una gran cantidad de datos en tiempo real y los resultados de los experimentos de estabilidad en el mar. Este método utiliza la estadística para obtener la fórmula empírica para calcular las características de entrada en pérdida de un buque en condiciones de viento y olas. El tercer método es el más utilizado en la práctica.

Su formulación matemática puede estudiarse en diversos artículos como Seuren,J, 2020 [2] , Wang et al., 2018 [3] o Zhao, W. et al., 2021 [14]; pero debido a su complejidad computacional y a la enorme cantidad de variables y datos a procesar para su cálculo se ha optado por la implementación de la metodología propuesta en M. Abebe et al., 2020 [10].

El artículo presenta una técnica basada en el aprendizaje automático para predecir la velocidad de un barco sobre el suelo (SOG) usando los datos del sistema de identificación automática (AIS) y los datos meteorológicos marítimos del mediodía. El SOG es la velocidad del barco en una hora con respecto a la tierra o cualquier otro objeto fijo, como boyas. El SOG se ve afectado por varios factores, como el viento, las olas, la corriente, la temperatura y la salinidad del agua, el estado de carga y el rumbo del barco.

Se estudio la aplicabilidad diferentes técnicas de regresión basadas en árboles de decisión para estimar el SOG a partir de las características seleccionadas de los datos del AIS y del clima. Además, compara el rendimiento de los modelos usando medidas de precisión como el coeficiente de determinación (R^2), el error cuadrático medio ($RMSE$) y el error cuadrático medio normalizado ($NRMSE$).

Algunos de los modelos empleados son la regresión lineal (LR), la regresión polinómica, los árboles de decisión (DTR), los métodos de ensamble como el bosque aleatorio (RFR), el XGBoost ($XGBR$) y el extra tree regressor (ETR). Estos modelos buscan establecer una relación entre el SOG y las variables operativas y ambientales que afectan el rendimiento del barco. El paper concluye que el modelo de regresión de árboles extra (ETR) es el más preciso y eficiente para predecir el SOG.

En este Trabajo de Fin de Grado, se pretende realizar un estudio de modelización del SOG de diferentes tipos de barcos, basándose en los métodos y resultados presentados en el paper de M. Abebe et al., 2020 [10]. Se utilizarán los mismos datos del AIS y los datos meteorológicos marinos para entrenar y validar los modelos predictivos, así como las medidas de precisión para evaluar su rendimiento. La adquisición de los datos, preprocesado, análisis e implementación de los modelos se describe en el Capítulo 6.

3.5.2. Modelos Predictivos

A continuación se describen las técnicas de modelado y el método general de los modelos de aprendizaje automático que se han utilizado para predecir el SOG de los barcos; como DTR, LR y modelos de conjunto (*ensemble models*), como ETR, RFR y $XGBR$.

3.5.2.1. Decision Tree Regresor (DTR)

El DTR es un método de regresión de aprendizaje supervisado no paramétrico [101] en forma de estructura de árbol con nodos y ramas. En el DTR, las características se dividen en un espacio rectangular y se entrena un modelo simple (árbol) para cada característica. Los modelos aprenden utilizando un

conjunto de datos de entrenamiento en un rango continuo. Su salida acaba siendo el valor medio de las observaciones de los conjuntos de entrenamiento que se encuentran en el mismo nodo. Los árboles de clasificación y regresión (CART) son uno de los métodos más comunes para los métodos de regresión basados en árboles. En CART, el espacio de características se dividirá en dos regiones tras elegir el punto de división óptimo para obtener el mejor ajuste del modelo. Esto se ejecutará recursivamente hasta que se activen las reglas de parada [10].

Para desarrollar el modelo, para un número n dado de muestras de datos y un número d de características, $D\{(\mathbf{x}_i, y_i)\}$ $\left(|D| = n, \quad \mathbf{x}_i \in R^d, y_i \in R\right)$ se supone que el espacio de características se divide en un número K de regiones, denominadas R_K y el valor de predicción del modelo se obtiene a partir del valor medio de la observación que se encuentra en la región k^{th} :

$$\hat{y}_i = \boldsymbol{ave}\left(y_i \mid \mathbf{x}_i \in R_k\right) \tag{3.17}$$

El mejor $\hat{y}_i$ se puede obtener minimizando el error cuadrático mínimo de $\sum (y_i - \bar{y}_i)^2$. Aunque los valores óptimos $\hat{y}_i$ se puede calcular de forma sencilla, sin embargo, no es fácil de dividir la región. Para superarlo, se aplica recursivamente un algoritmo *greedy* para determinar los nodos de división óptimos hasta llegar al punto de parada. Normalmente, esto depende de los hiperparámetros y de la dificultad del problema [10].

3.5.2.2. Linear Regressor (LR)

La regresión lineal es un algoritmo de aprendizaje supervisado para predecir valores continuos a partir de variables de entrada. Este algoritmo establece una relación lineal entre las variables independientes (variables de entrada, características o predictores) y la variable dependiente (variable de salida o variable objetivo) [102].

El algoritmo encuentra la línea de mejor ajuste que minimiza la suma de errores al cuadrado entre los valores predichos y los valores reales. Esta línea se denomina línea de regresión o línea de mejor ajuste. La ecuación de esta recta es de la forma

$$y = \beta 0 + \beta 1 * x1 + \beta 2 * x2 + \ldots + \beta n * xn \tag{3.18}$$

donde y es la variable dependiente, $x_1, x_2, \ldots, x_n$ son las variables independientes, $\beta 0$ es el intercepto, y $\beta_1, \beta_2, \ldots, \beta_n$ son los coeficientes [101].

El objetivo del algoritmo de Regresión Lineal es estimar los valores de estos coeficientes ($\beta_0, \beta_1, \beta_2$, $\ldots, \beta_n$) de forma que se minimice la suma de errores al cuadrado. Este proceso se denomina método de mínimos cuadrados ordinarios (MCO).

3.5.2.3. Métodos de conjunto (*Ensemble Methods*)

Los métodos de aprendizaje de conjunto consisten en desarrollar un modelo de predicción integrando varios modelos simples. Los dos métodos más comunes son el *bagging* y el *boosting*. El *bagging* combina varios modelos base en uno solo para generar un modelo ensamble más robusto, por ejemplo, mediante el promedio de regresiones. El *bagging* reduce la varianza y se puede aplicar para modelos con alta varianza y bajo sesgo. El *boosting*, en cambio, genera un modelo ensamble a partir de un solo modelo, como los árboles de decisión. Es una técnica secuencial que combina un conjunto de aprendices débiles y proporciona una estimación más precisa. El *boosting* produce modelos fuertes con bajo sesgo. Las nuevas salidas del modelo desarrollado tienen pesos basados en las salidas anteriores del modelo. Si las salidas se predicen correctamente, se asigna un peso menor; de lo contrario, el peso asignado es mayor [10].

3.5.2.3.1. Random Forest Regressor (RFR)

RFR es un método de regresión basado en aprendizaje automático que se propuso por Breiman [103] y se desarrolló a partir de la técnica de bagging. El modelo RFR se construye generando varios árboles de decisión decorrelacionados (n_estimadores) usando el conjunto de entrenamiento. La respuesta del modelo RFR se obtiene promediando los resultados de los árboles individuales:

$$\hat{y}_i(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} f_m(\mathbf{x}_i) \tag{3.19}$$

donde M es el número de árboles (n_estimadores). Para construir un árbol, el método usa una réplica bootstrap del conjunto de entrenamiento (muestreo con reemplazamiento) y el algoritmo CART. Un óptimo corte sobre una submuestra de características en cada nodo de prueba se obtiene buscando una submuestra aleatoria con el tamaño de las características candidatas. Esto significa que se selecciona una submuestra sin reemplazo de las características candidatas con el menor tamaño de muestra para dividir el nodo [10].

3.5.2.3.2. Extra Trees Regressor (ETR)

El algoritmo ETR es una técnica de aprendizaje automático que desarrolla un conjunto de árboles de regresión sin podar, utilizando un enfoque de crecimiento de arriba hacia abajo. La diferencia entre ETR y RFR es que los puntos de corte seleccionados de los nodos de división en ETR son extremadamente aleatorios para crecer el árbol, además, ETR usa todo el conjunto de entrenamiento en lugar de una réplica bootstrap. En términos de sus características numéricas, el proceso de división en ETR se rige por dos parámetros clave: el número de características seleccionadas al azar en cada nodo y el tamaño mínimo de muestra necesario para dividir un nodo. Para obtener el resultado final, ETR combina los modelos predictivos generados por los árboles individuales, similar a lo que ocurre en el caso de RFR. Los modelos predictivos se combinan de manera ponderada para producir el resultado final de la predicción, por ejemplo, mediante el promedio en problemas de regresión [10].

3.5.2.3.3. Extreme Gradient Boosting Regressor (XGBR)

El refuerzo de gradiente se refiere a una clase de algoritmos de aprendizaje automático de conjunto que pueden utilizarse para problemas de modelado predictivo de clasificación o regresión.

Los conjuntos de modelos se construyen a partir de árboles de decisión, los cuales se añaden secuencialmente al conjunto y se ajustan para corregir los errores de predicción cometidos por los modelos anteriores. Este tipo de modelo se conoce como boosting en el ámbito del aprendizaje automático.

El proceso de ajuste de los modelos se realiza utilizando una función de pérdida diferenciable y un algoritmo de optimización basado en el descenso de gradiente. De ahí el nombre de la técnica, "refuerzo por gradiente", ya que el gradiente de la función de pérdida se minimiza gradualmente a medida que se ajusta el modelo, de manera similar a lo que ocurre en una red neuronal [104].

XGBR son un tipo de refuerzo de gradiente optimizado y distribuido, que se diseñan para ser eficientes, flexibles y portátiles [105]. XGBR proporciona hiperparámetros de regularización adicionales como se muestra en la ecuación 3.20, que pueden ayudar a reducir las posibilidades de sobreajuste, disminuir la variabilidad de la predicción y, por tanto, mejorar la precisión. El valor predicho $\hat{y}_i$ se obtiene minimizando la función de regulación L :

$$L = \sum_i l\left(\hat{y}_i, y_i\right) + \sum_m \Omega\left(f_m\right), \text{ donde } \Omega(f) = \gamma T + \frac{1}{2}\lambda\|w\|^2 + \alpha|w| \tag{3.20}$$

Aquí, Ω representa el parámetro de regularización que penaliza la complejidad del modelo como funciones de árbol de regresión y suaviza los pesos finales aprendidos para evitar el sobreajuste. T representa el número de nodos hoja y w es la puntuación del nodo hoja. γ, λ, y α se usan para definir el nivel de regularización. α y λ también conocidos como regularización L_1 y L_2, respectivamente, tienen diferentes influencias en el peso; α se asocia la dispersión, "alentando" al peso a ser cero, mientras que λ "hace tender" al peso a ser pequeño. γ es un hiperparámetro de pseudo-regularización comúnmente implementado conocido como un multiplicador de Lagrange que controla la complejidad de un árbol dado. γ especifica la reducción mínima de pérdida requerida para hacer más particiones en un nodo hoja, lo que significa que un valor más alto conduce a menos divisiones [10].

El proceso de predicción suma los resultados de cada árbol para obtener los resultados finales en el modelo XGBR. Los parámetros de cada árbol (f_t), que incluyen la estructura del árbol y las puntuaciones obtenidas por cada nodo hoja, tienen que ser determinados. El método de entrenamiento aditivo agrega el resultado de un árbol al modelo en un momento dado. El valor predicho $\left(\hat{y}_i^{(t)}\right)$ obtenido en el paso t se puede usar para obtener el proceso del algoritmo [10]:

$$\hat{y}_i^{(t)} = \sum_{m=1}^{M} f_m\left(\mathbf{x}_i\right) = \hat{y}_i^{(t-1)} + f_t\left(\mathbf{x}_i\right) \tag{3.21}$$

3.5.3. Análisis estadístico

Para asegurar la precisión de los modelos planteados, se utilizan diversos estadísticos comúnmente utilizados como los siguientes [10]:

3.5.3.1. Error medio absoluto (MAE)

Representa la diferencia entre los valores originales y los valores predichos, cuyo cálculo se obtiene del promedio de las diferencias absolutas de todo el conjunto de valores. Su formulación es la siguiente:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} x_i - \hat{x}_i \tag{3.22}$$

donde $\hat{x}$ es el valor predicho del modelo y x_i el dato observado.

3.5.3.2. Error cuadrático medio (MSE)

Representa la diferencia entre los valores originales y los valores predichos, calculada mediante el cuadrado de la diferencia promedio de todo el conjunto de valores. Su formulación es la siguiente:

$$MSE = \sum_{i=1}^{n}(x_i - \hat{x}_i)^2 \tag{3.23}$$

donde $\hat{x}$ es el valor predicho del modelo y x_i el dato observado.

3.5.3.3. Coeficiente de determinación (R^2)

Se trata de una medida importante a la hora de evaluar la precisión del modelo para análisis de regresión, representando cómo de bien los valores se ajustan a los valores originales. Se expresa como la proporción de la varianza de la característica dependiente predicha con respecto a la característica independiente. El coeficiente de determinación se define como:

$$R^2 = 1 - \frac{\sum (x_i - \hat{x})^2}{\sum (x_i - \bar{x})^2} \tag{3.24}$$

donde $\hat{x}$ es el valor predicho del modelo, x_i el dato observado y $\bar{x}$ es la media de los datos observados.

La escala de R^2 varía de 0 a 1; 0 indica que el modelo propuesto no mejora la predicción en comparación con el modelo medio, y 1 indica una predicción perfecta.

3.5.3.4. Error cuadrático medio ($RMSE$)

El error cuadrático medio ($RMSE$) es la raíz cuadrada de la varianza de las diferencias individuales, también conocidas como residuos. Indica qué tan cerca están los valores del modelo de estimación de los valores de los datos observados. En general, $RMSE$ es una medida absoluta de la calidad de ajuste de un modelo, mientras que R^2 es una medida relativa de ajuste. Un valor más bajo de $RMSE$ indica un mejor ajuste. $RMSE$ se define de la siguiente manera:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2} \tag{3.25}$$

donde $\hat{x}$ es el valor predicho del modelo y x_i el dato observado.

Capítulo 4

Análisis

Aunque el presente TFG se centra en el estudio y elaboración de algoritmos genéticos que permitan la optimización de las rutas marítimas, se ha decidido realizar una pequeña aplicación web que permita interactuar con el modelo creado y mostrar los resultados obtenidos con las rutas especificadas y sus características.

4.1. Análisis de requisitos

Desde el comienzo del proyecto hasta en cada historia de usuario desarrollada, se han desglosado los requisitos del proyecto en forma de historias de usuario.

Las historias de usuario ofrecen una descripción sobre la funcionalidad deseada, en la que lo importante es hablar sobre los requisitos más allá de el enfoque de escribirlos. Estas historias adquieren la perspectiva de la persona interesada en la nueva función con la estructura: Como **Usuario** ... , Quiero **algún objetivo** para que **motivo**.

Para la completitud de estas, las historias de usuario vienen acompañadas con los criterios de aceptación . Los criterios de aceptación son pruebas a alto nivel que se cumplirán después de que se completa la historia de usuario [106].

4.1.1. Requisitos funcionales como historias de usuario

Código	Historia de usuario
HU01	Como usuario quiero obtener las rutas óptimas en un mapa para llegar desde el punto de salida a un punto de llegada
HU02	Como usuario quiero poder analizar las rutas óptimas en una tabla u otra forma que muestre información detallada sobre la distancia, el tiempo estimado, la velocidad del barco y cualquier otra información relevante que pueda ayudarme a planificar mi viaje en el mar.
HU03	Como usuario quiero poder ingresar la longitud y la latitud de mi punto de partida, así como la fecha de salida, para que el algoritmo pueda construir la ruta marítima de mi barco.
HU04	Como usuario, quiero poder modificar los parámetros que hacen funcionar el algoritmo para poder ajustar el algoritmo y el resultado de este.
HU05	Como usuario, quiero poder modificar los parámetros de la embarcación que toma el algoritmo, para que el algoritmo de navegación pueda tener en cuenta estos factores y crear una ruta óptima para mi barco.

Tabla 4.1: Requisitos funcionales

Código	Criterios de aceptación
HU01	▪ El sistema debe mostrar la ruta óptima en el mapa, teniendo en cuenta la velocidad del barco, la dirección del viento y las corrientes marinas. ▪ La ruta óptima debe ser clara y fácil de entender para el usuario. ▪ El sistema debe ser capaz de calcular la ruta óptima en un tiempo razonable.
HU02	▪ El sistema debe ser capaz de mostrar los resultados de cada ruta óptima obtenida en una tabla. ▪ La tabla debe mostrar información detallada sobre la distancia, el tiempo estimado, la velocidad del barco y cualquier otra información relevante. ▪ El usuario debe ser capaz de comparar fácilmente los resultados de cada ruta en la tabla. ▪ La tabla debe ser clara y fácil de entender para el usuario. ▪ El sistema debe ser capaz de calcular y mostrar los resultados de las rutas en un tiempo razonable.
HU03	▪ El sistema debe permitir al usuario ingresar la longitud y la latitud de su punto de partida y punto de llegada. ▪ El sistema debe permitir al usuario ingresar la fecha de salida deseada. ▪ El sistema debe utilizar la información de entrada proporcionada por el usuario para calcular la ruta marítima óptima. ▪ La ruta marítima calculada por el sistema debe tener en cuenta factores como la velocidad y la dirección del viento, las corrientes oceánicas y otras condiciones ambientales relevantes. ▪ El sistema debe presentar la ruta marítima calculada al usuario de manera clara y fácil de entender, proporcionando información como la distancia total del viaje, la duración estimada del viaje, y cualquier información adicional relevante para la navegación. ▪ El sistema debe permitir al usuario ajustar los parámetros de entrada en cualquier momento, para que pueda realizar cambios en la ruta marítima si es necesario. ▪ El sistema debe ser fácil de usar y tener una interfaz intuitiva que permita al usuario ingresar la información de entrada de manera rápida y sencilla.

HU04	▪ El sistema debe permitir al usuario modificar los parámetros del algoritmo en cualquier momento. ▪ Los parámetros deben ser fáciles de entender y ajustar para el usuario, con una interfaz clara e intuitiva. ▪ Al modificar los parámetros, el algoritmo debe ajustar los nuevos resultados a los nuevos parámetros. ▪ El sistema debe ser capaz de manejar una amplia variedad de parámetros diferentes, y permitir al usuario ajustarlos de forma individual o en conjunto. ▪ El sistema debe ser capaz de manejar los ajustes en tiempo real, sin retrasos notables en el tiempo de respuesta.
HU05	▪ El usuario debe ser capaz de modificar los parámetros de longitud, calado y anchura de su barco. ▪ La nueva ruta generada debe tomar en cuenta los parámetros modificados por el usuario para ofrecer una ruta óptima para el barco. ▪ La aplicación debe permitir al usuario revertir/modificar los cambios realizados a los parámetros del barco en cualquier momento.

Tabla 4.2: Criterios de aceptación requisitos funcionales

4.1.2. Requisitos no funcionales como historias de usuario

Código	Historia de usuario
HUNF01	Como usuario, quiero que el proyecto se materialice en una aplicación web, para poder acceder a la funcionalidad de cálculo de rutas desde cualquier dispositivo con conexión a internet, y poder visualizar las rutas en un mapa interactivo
HUNF02	Como usuario del sistema, quiero que la interfaz de usuario sea intuitiva junto con un diseño atractivo, de modo que pueda utilizar el sistema sin dificultades y obtener la ruta sin problemas
HUNF03	Como usuario de una aplicación web de navegación marítima, quiero que el algoritmo de cálculo de rutas sea rápido y eficiente, para que pueda obtener las rutas óptimas de navegación en el menor tiempo posible
HUNF04	Como equipo de desarrollo, queremos implementar un algoritmo genético para la planificación de rutas de barcos, para que la solución sea escalable, eficiente y de alta calidad.

Tabla 4.3: Requisitos no funcionales

Código	Criterios de aceptación
HUNF01	▪ La aplicación web debe contar con una interfaz gráfica intuitiva, que permita al usuario introducir los parámetros necesarios para el cálculo de rutas de manera sencilla y eficiente. ▪ La aplicación web debe contar con un mapa interactivo que muestre las rutas óptimas de navegación, permitiendo al usuario visualizar la ruta en tiempo real, acercar y alejar el mapa, y desplazarse por el mismo. ▪ La aplicación web debe permitir al usuario modificar los parámetros del barco y del algoritmo para que el algoritmo de cálculo de rutas pueda proporcionar las rutas óptimas de navegación. ▪ La aplicación web debe proporcionar al usuario una lista de las rutas óptimas de navegación, teniendo en cuenta el tiempo de llegada, fuel consumido, para que el usuario pueda seleccionar la ruta que mejor se adapte a sus necesidades. ▪ El equipo de desarrollo debe realizar pruebas de rendimiento para verificar que la aplicación web pueda soportar un alto tráfico de usuarios sin afectar su velocidad de carga, precisión en el cálculo de rutas y funcionalidad del mapa interactivo.

HUNF02	▪ La interfaz de introducción de parámetros debe ser clara y fácil de entender para el usuario, con una estructura lógica y coherente en todo el sistema. ▪ Los campos de entrada de los parámetros deben ser intuitivos y estar ubicados de manera coherente en todas las pantallas de la interfaz. ▪ Los parámetros del barco, ruta y algoritmo ingresados por el usuario deben ser validados para asegurar que sean valores aceptables y que cumplan con los requisitos del sistema. ▪ El algoritmo de navegación debe tener en cuenta los parámetros del barco ingresados por el usuario para generar una ruta óptima que considere las condiciones actuales del mar, la distancia a recorrer y los tiempos estimados de viaje. ▪ La ruta generada debe ser presentada al usuario en un mapa interactivo con las rutas óptimas y las posibles alternativas de ruta, junto con información detallada sobre la distancia y tiempo estimado para cada opción. ▪ La interfaz del mapa debe ser clara y fácil de entender para el usuario, con una navegación sencilla y coherente. ▪ La carga de la página web y la presentación de la información en el mapa deben ser rápidas y eficientes, para asegurar una experiencia de usuario fluida y sin demoras.
HUNF03	▪ El algoritmo de cálculo de rutas debe ser lo suficientemente rápido para proporcionar las rutas óptimas de navegación en un tiempo razonable, no superando diez minutos. ▪ La precisión del cálculo de las rutas debe ser alta, asegurando que se tomen en cuenta los parámetros del barco y las condiciones meteorológicas para proporcionar las rutas óptimas de navegación. ▪ La aplicación web debe mostrar un mensaje de espera o barra de progreso mientras se está realizando el cálculo de las rutas, para que el usuario pueda saber que la página está trabajando y esperar pacientemente el resultado. ▪ El equipo de desarrollo debe realizar pruebas de rendimiento para verificar que el algoritmo pueda procesar un alto volumen de solicitudes sin afectar su velocidad y precisión en el cálculo de las rutas.
HUNF04	▪ El algoritmo debe ser eficiente y escalable para manejar grandes conjuntos de datos y garantizar un rendimiento óptimo en todo momento. ▪ El algoritmo debe ser probado y validado para asegurar su correcto funcionamiento y rendimiento en diferentes situaciones y casos de uso. ▪ El código debe estar documentado y seguir los estándares de calidad y estilo de codificación del equipo de desarrollo.

Tabla 4.4: Criterios de aceptación requisitos no funcionales

4.1.3. Requisitos de información como historias de usuario

Historia de usuario
Como equipo de desarrollo, quiero almacenar un modelo que agrupa las características de las condiciones del mar y las señales AIS de los barcos , para poder utilizarlo en un algoritmo de planificación de rutas de barcos y obtener rutas óptimas.

Tabla 4.5: Requisitos de información

Criterios de aceptación
▪ El modelo de datos debe incluir la información de CMEMS physics and wave data y AIS data en un formato que sea fácil de utilizar en el algoritmo. ▪ El modelo de datos debe ser almacenado en un lugar seguro y confiable, con medidas de seguridad adecuadas para proteger los datos. ▪ El modelo de datos debe ser actualizado regularmente para mantenerlo al día y garantizar su precisión. ▪ El algoritmo de planificación de rutas debe ser capaz de utilizar el modelo de datos almacenado para calcular rutas óptimas en función de las condiciones del mar y las señales AIS de los barcos.

Tabla 4.6: Criterios de aceptación requisitos de información

4.1.4. Reglas de negocio como historias de usuario

Historia de usuario
Como equipo de desarrollo, se extrerán los datos de CMEMS physics y wave data utilizando los protocolos de OpenDAP y/o FTP para garantizar la integridad y la calidad de los datos.

Tabla 4.7: Reglas de negocio

Criterios de aceptación
▪ La extracción de datos debe realizarse utilizando los protocolos de OpenDAP y/o FTP. ▪ Los datos extraídos deben ser validados y comprobados para garantizar la integridad y la calidad de los datos. ▪ Se debe proporcionar una documentación clara y detallada sobre el proceso de extracción de datos para permitir su replicabilidad y transparencia. ▪ Se deben establecer procedimientos de control de calidad para verificar la precisión y la calidad de los datos extraídos.

Tabla 4.8: Criterios de aceptación reglas de negocio

4.2. Modelo de dominio

La Fig. 4.1 muestra el modelo del dominio que representa de manera abstracta las clases que definen, en una primera instancia, nuestro sistema, con los atributos (datos) que las constituyen y las relaciones entre ellas. La cardinalidad de las conexiones indica la cantidad de relaciones que pueden existir entre las entidades o clases.

Este modelo ha sido elaborado a partir de los requisitos funcionales, en modo historias de usuario, de información mencionados en la tabla **4.1** y los criterios de aceptación **4.2**.

La estructuración del modelo inicial que hemos seguido ha tratado de cubrir los siguientes aspectos:

- **Puerto:** Identificar el puerto de salida y de llegada de la ruta propuesta a seguir por el barco junto a los parámetros de coordenadas que los identifican.

- **Barco:** Con las características a tener en cuenta para que se adapte la ruta óptima.

- **Ruta:** Con los tiempos y valor final de los objetivos de la ruta óptima.

- **AlgoritmoGenetico:** Con sus parámetros de ajuste de selección,mutación y cruce calcula la ruta óptima teniendo en cuenta el Barco, el tiempo y puntos de la ruta y las condiciones climatológicas y marítimas.

- **Condiciones:** Condiciones climatológicas y marítimas para esa ruta en ese espacio temporal.

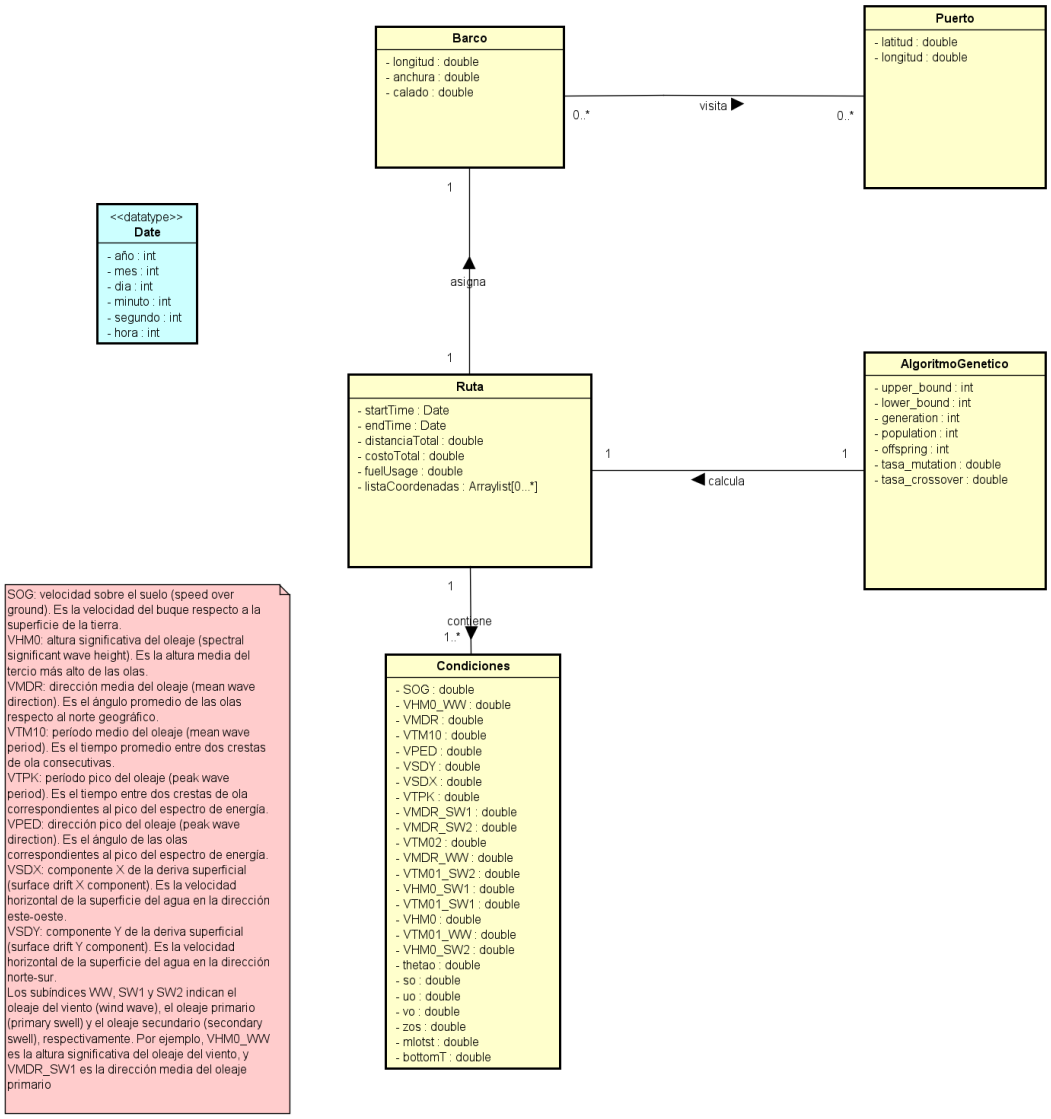

Figura 4.1: Diagrama de modelo de dominio

4.3. Casos de uso

En esta Sección se presentan los casos de uso que forman parte del sistema para llevar a cabo el proyecto. Estos nos permiten una mejor comprensión de la interacción entre los actores y nuestro sistema. Por lo que en los siguientes apartados se muestra el diagrama de realización en análisis y la descripción de los mismos, además, de los diagramas de secuencia y flujo que aportan mayor nivel de detalle en la interacción y los pasos que se realizan.

4.3.1. Actores Principales

Hasta ahora, se ha identificado únicamente a un actor principal en la ecuación: el usuario que utiliza la aplicación.

- **Usuario App**: Aquel que interactúa con el sistema para acceder a la información que requiere (rutas marítimas óptimas para unos parámetros concretos).

4.3.2. Diagrama y Descripción de los casos de uso

En la Fig. 4.2 se muestra el diagrama de los casos de uso desarrollados en el sistema, junto al actor mencionado anteriormente. Además se especifican los diferentes casos de uso con una identificador, nombre, descripción, actores, precondición, secuencia normal, postcondición y secuencia alternativa.

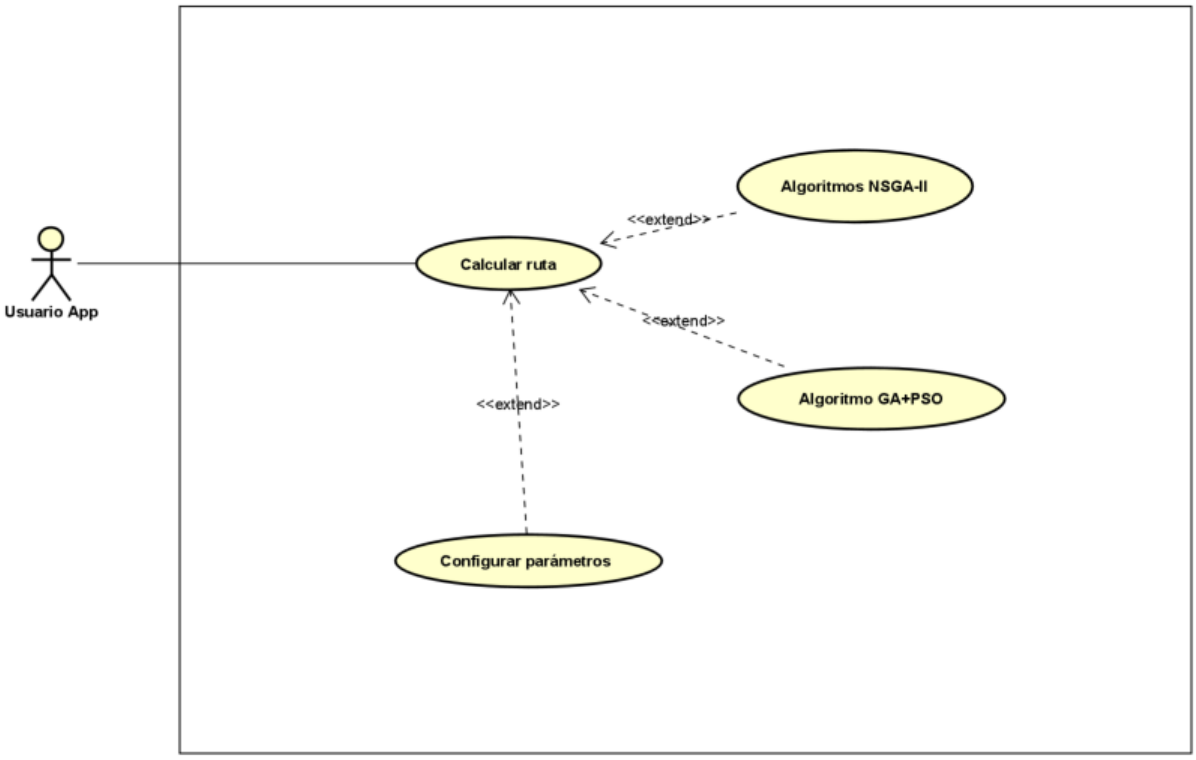

Figura 4.2: Diagrama de modelo de casos de uso

CU-01	**Calcular Ruta**
Descripción	El usuario desea visualizar las rutas marítimas óptimas según sus parámetros.
Actores	Usuario App
Precondición	La aplicación necesita estar activa en el servidor y el usuario ha ingresado a ella mediante la URL del sitio web.
Secuencia Normal	1. El usuario introduce los datos del barco. 2. El usuario introduce los datos de la ruta. 3. El usuario selecciona la opción de Ejecutar Algoritmo con los parámetros prefijados. 4. Se realiza el caso de uso <Algoritmo NSGA-II> 5. El sistema muestra los resultados obtenidos.
Postcondición	El sistema muestra en un mapa los resultados (rutas) obtenidas.
Secuencia Alternativa	1.a,2.a - Se produce un error al procesar los datos de entrada del usuario, se devuelve un error y el caso de uso queda sin efecto 3.a - Se realiza el caso de uso <Configurar Parámetros>. 4.b -Se realiza el caso de uso <Algoritmo GA+PSO>.

Tabla 4.9: Caso de Uso 1- Calcular Ruta

CU-02	**Algoritmo NSGA-II**
Descripción	El sistema debe permitir aplicar el algoritmo NSGA-II al cálculo de las rutas marítimas óptimas a partir de los parámetros proporcionados previamente.
Actores	Usuario App
Precondición	El usuario debe haber introducido de manera correcta los parámetros a tener en cuenta en el algoritmo, y el CU-01 ha sido iniciado.
Secuencia Normal	1. El sistema obtiene las rutas marítimas óptimas mediante el Algoritmo NSGA-II con los parámetros proporcionados. 2. El sistema devuelve las rutas obtenidas.
Postcondición	
Secuencia Alternativa	1.a,2.a - Se produce un error en la ejecución del algoritmo, se devuelve un error y el caso de uso queda sin efecto.

Tabla 4.10: Caso de Uso 2- Algoritmo NSGA-II

CU-03	**Algoritmo GA+PSO**
Descripción	El sistema debe permitir aplicar el algoritmo GA+PSO al cálculo de las rutas marítimas óptimas a partir de los parámetros proporcionados previamente.
Actores	Usuario App
Precondición	El usuario debe haber introducido de manera correcta los parámetros a tener en cuenta en el algoritmo, y el CU-01 ha sido iniciado.
Secuencia Normal	1. El sistema obtiene las rutas marítimas óptimas mediante el Algoritmo GA+PSO con los parámetros proporcionados. 2. El sistema devuelve las rutas obtenidas.
Postcondición	
Secuencia Alternativa	1.a,2.a- Se produce un error en la ejecución del algoritmo, se devuelve un error y el caso de uso queda sin efecto.

Tabla 4.11: Caso de Uso 3- Algoritmo GA+PSO

CU-04	**Configurar parámetros**
Descripción	El sistema debe permitir al usuario modificar los parámetros prefijados para los algoritmos, de forma individual o en su conjunto.
Actores	Usuario App
Precondición	El CU-01 ha sido iniciado.
Secuencia Normal	1. El usuario modifica los parámetros prefijados 2. El sistema utiliza los nuevos parámetros para poder ejecutar el algoritmo con ellos.
Postcondición	
Secuencia Alternativa	1.a- Se produce un error a la hora de introducir los nuevos datos, se devuelve un error y el caso de uso queda sin efecto.

Tabla 4.12: Caso de Uso 4- Configurar parámetros.

4.3.3. Realización en análisis de los casos de uso

A continuación se presenta el diagrama de secuencia del caso uso principal Calcular Ruta, según ha sido descrito anteriormente para poder apreciar la interacción entre el actor y el sistema.

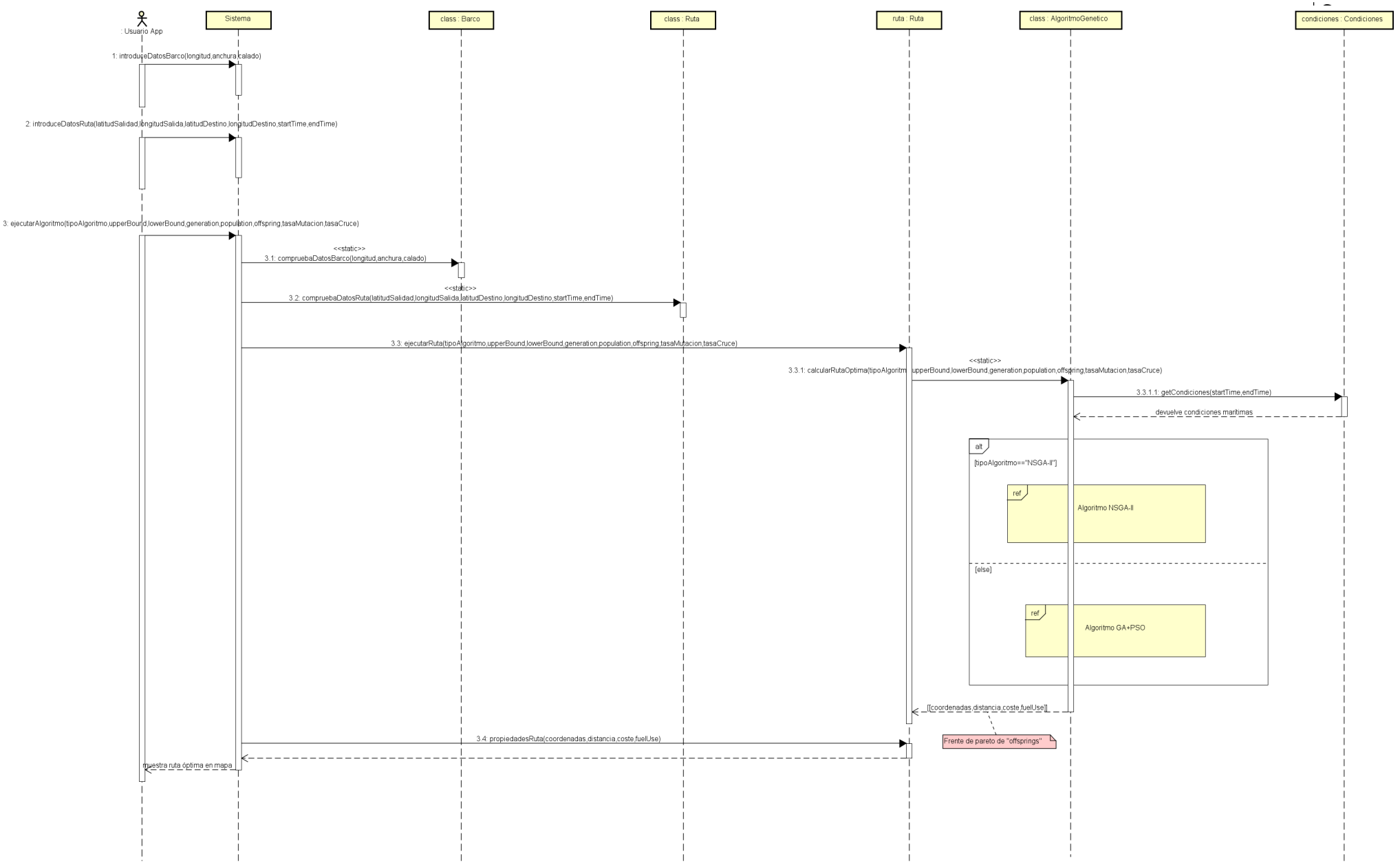

Figura 4.3: Realización en análisis del caso de uso Calcular Ruta.

4.4. Diagramas de flujo

Para completar el análisis del sistema, se ha llevado a cabo un diagrama de flujo por cada algoritmo planteado, en la que se represente los diferentes pasos que se atraviesa al ejecutarlos. Se sigue la descripción algorítmica y de flujo descrita en la Sección 3.4.

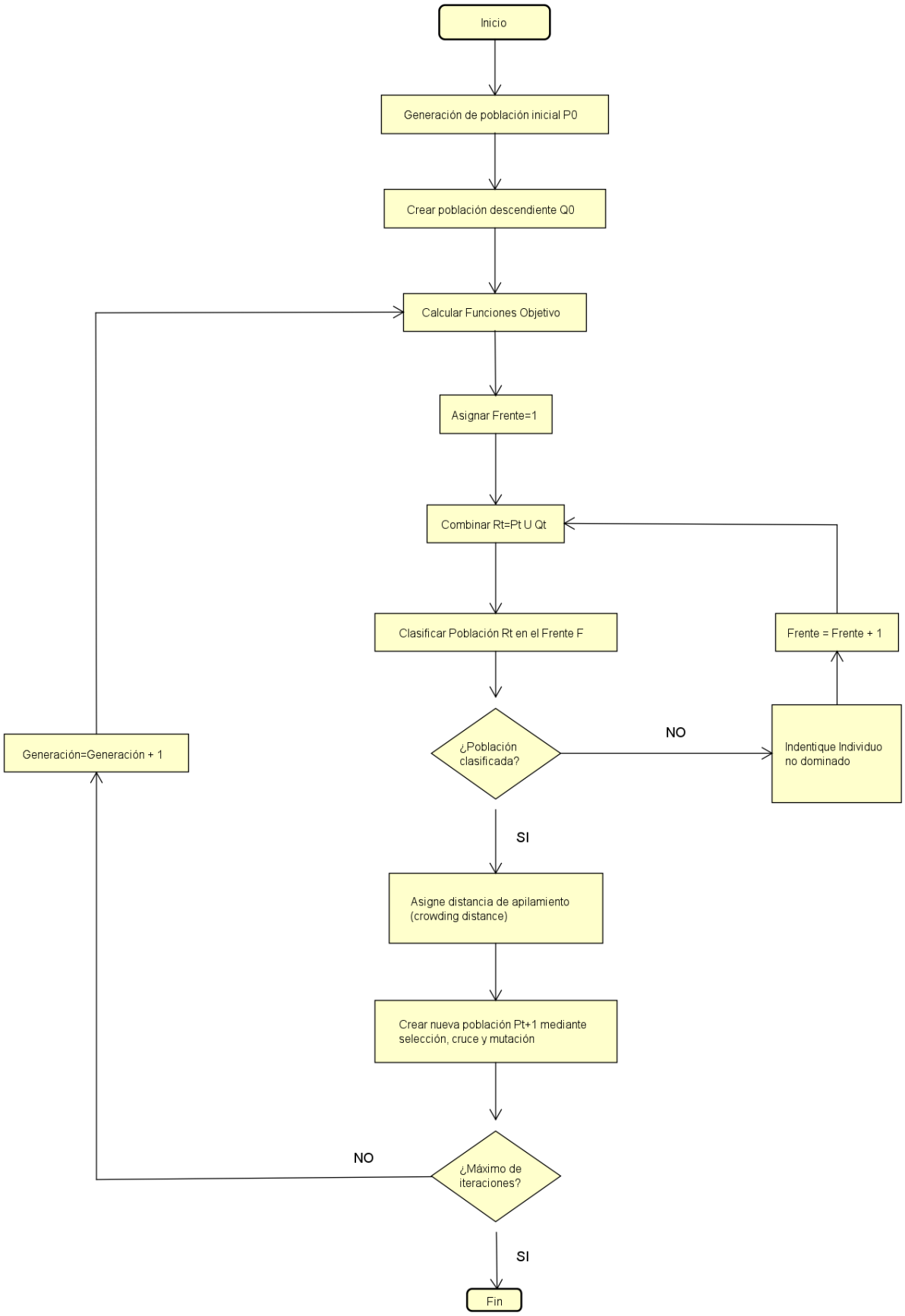

Figura 4.4: Diagrama flujo algoritmo NSGA-II

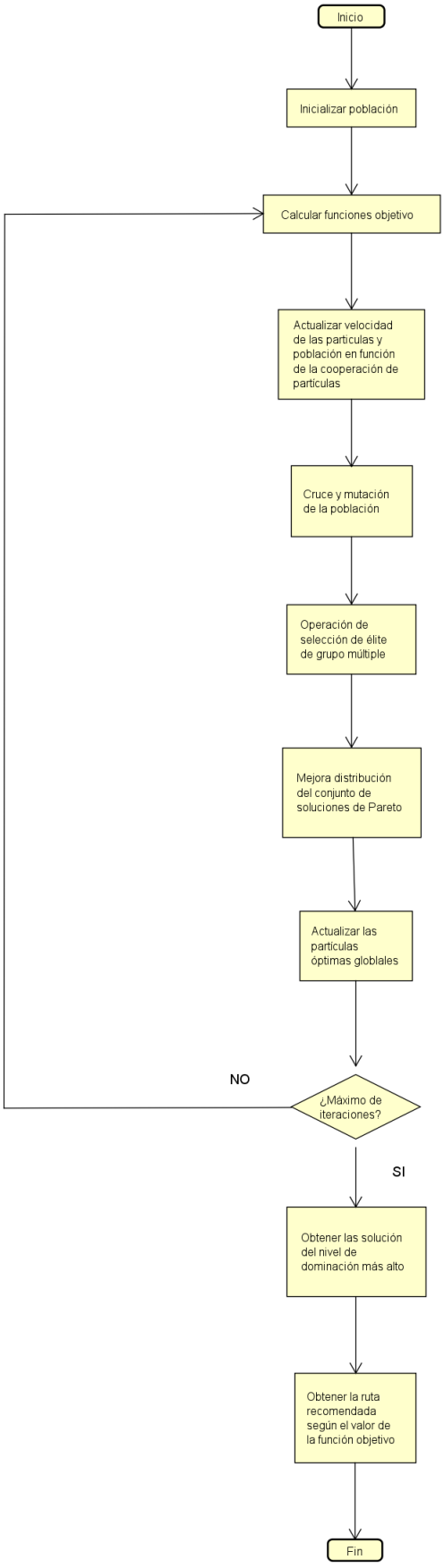

Figura 4.5: Diagrama flujo algoritmo GA+PSO

Capítulo 5

Diseño

Una vez tomado el paso de Análisis y antes de la fase de Implementación, se procede a describir las decisiones de diseño de la aplicación para la visualización de la ruta generada por el algoritmo genético. En las siguientes secciones se describirán las decisiones de diseño adoptadas, la arquitectura utilizada, diseño de la arquitectura del algoritmo genético y el diseño de la interfaz de la aplicación.

5.1. Decisiones de diseño

Se presentan las decisiones tomadas a la hora de la utilización de diversas tecnologías y patrones.

- Se usará una arquitectura cliente-servidor.
- En el lado del cliente, la aplicación se encargará de la parte gráfica como de los datos de la ruta, barco y puerto que introducirá el usuario.
- En el lado del servidor, se encargará de procesar y atender las peticiones a la API por parte del cliente. Para ello se encargará de descargar los datos meteorológicos del Servidor de Copernicus, modelarlos para que el algoritmo pueda calcular las rutas óptimas a devolver.
- Se utilizará Python a la hora de crear el algoritmo genético, apoyándonos en la librería Pymoo, y a la hora de extraer, modelar y preparar los datos meterológicos.
- Para el servidor se utilizará el framework Flask que depende de la especificación WSGI de Werkzeug y el motor de templates Jinja.
- Para el cliente se utilizará HTML puro junto con Bootstrap y CSS para el estilo.
- Para la representación de la ruta y los datos meteorológicos, se empleará la librería Plotly.js, que permite la creación de mapas interactivos con una alta configuración.

5.2. Patrones Arquitectónicos

En esta Sección se detallan los patrones arquitectónicos utilizados para la estructura del proyecto. En particular, se abordan el patrón cliente-servidor y el Modelo-Vista-Controlador.

5.2.1. Patrón Cliente-Servidor

El patrón cliente-servidor es un modelo de aplicación distribuida en el cual se reparten las responsabilidades y funciones entre los proveedores de recursos o servicios, llamados servidores, y los que solicitan dichos recursos, conocidos como clientes. Los clientes envían las peticiones al servidor y esperan una respuesta, mientras que el servidor procesa dichas solicitudes y envía las respuestas de vuelta a los clientes.

5.2.2. Patrón MVC

El patrón modelo-vista-controlador (MVC) se trata de un patrón arquitectónico de software que establece una separación entre los datos y la lógica de negocio, y la interfaz de usuario junto con el módulo responsable de gestionar los eventos y comunicaciones.

- El modelo: Representa la información con la que el sistema opera,
- La vista: Presenta la información del modelo en un formato adecuado para interactuar.
- El controlador: Responde a los eventos e invoca peticiones al modelo cuando se necesitas alguna información, haciendo de intermediario entre la vista y el modelo.

El MVC favorece la modularidad, reutilización y mantenimiento del código; como la separación de responsabilidades y el principio de bajo acoplamiento y alta cohesión.

En Flask, aunque no impone una forma de implementación con un patrón concreto, este patrón se adapta al framework, y posteriormente se describirá el uso del patrón junto a Flask.

5.3. Patrones de Diseño

Se detalla el patrón utilizado en diseño a la hora de implementación del algoritmo genético en la aplicación. Se hace uso del patrón estrategia que se define a continuación.

5.3.1. Patrón Estrategia

El patrón estrategia es un patrón de comportamiento que permite definir una serie de objetos diferentes correspondientes a la misma clase puedan elegir un comportamiento específico, en nuestro caso un algoritmo. El patrón estrategia encapsula los diferentes algoritmos en una jerarquía y permite que el objeto cliente interactúe con un objeto intermedio contexto [107].

La finalidad del patrón es desvincular los algoritmos de las clases que los utilizan, favoreciendo la reutilización y la mantenibilidad del código.

Este patrón se ha usado a la hora de crear las diferentes partes del algoritmo genético, junto a la librería Pymoo 8.1.6.

5.4. Arquitecturas del sistema

Como se ha comentado anteriormente, la aplicación se dividirá en cliente - servidor, tal como podemos observar en la Fig. 5.1; en la que el servidor a través de la herramienta Flask , ofrece una API Rest a la que el cliente realiza peticiones *HTTP* para obtener las rutas óptimas. Para ello el servidor establece conexión y hace una consulta al servidor de Copernicus mediante *OPeNDAP* para obtener los datos meteorológicos y marinos.

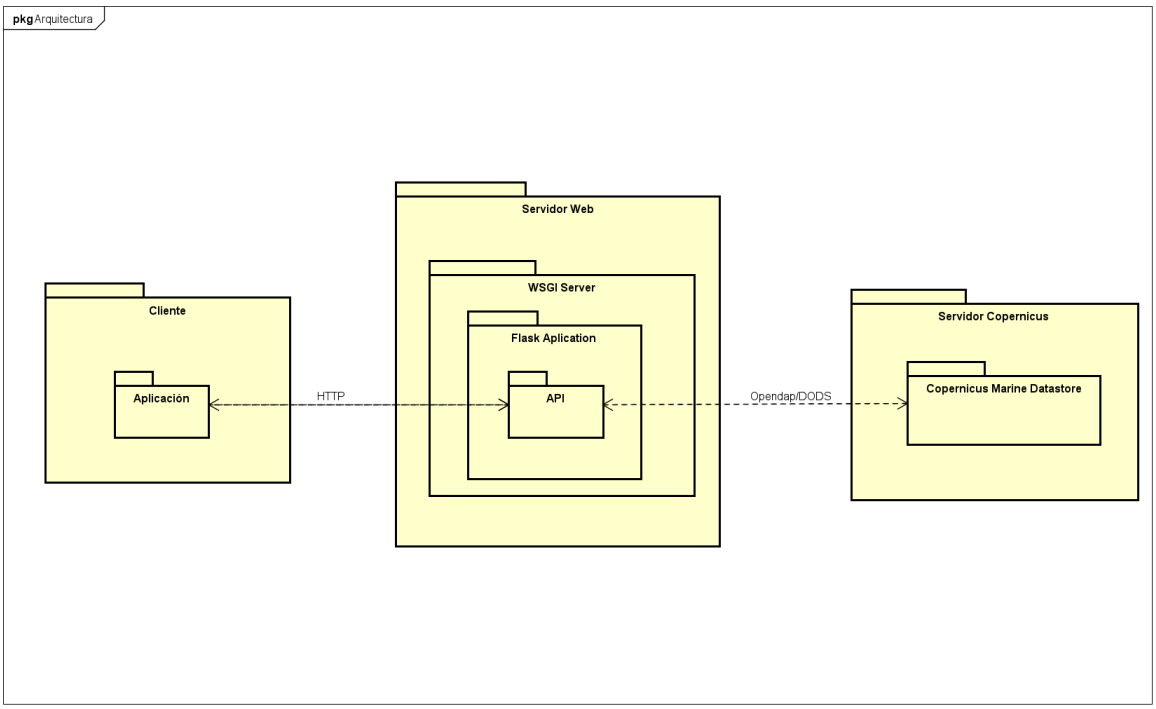

Figura 5.1: Arquitectura Lógica del sistema.

5.5. Arquitectura del servidor

Como se ha comentado anteriormente, el servidor utiliza el framework de Flask, que es el que nos sirve la API y atiende las peticiones del cliente. Se ha desarrollado bajo el patrón arquitectónico *MVC* (Modelo-Vista-Controlador). En él y como se muestra en la Fig. 5.2, *Flask View Controller* asume el papel de Vista-Controlador en el framework Flask tal como se explicará en detalle en 5.5.1 y la carpeta modelo es la que asume la llamada al Servidor de Copernicus para recuperar los datos marítimos y hacer las transformaciones, cálculos y predicciones del SOG de estos para que pueda ser utilizado por el algoritmo. En la Fig. 5.3 se entra más aún en detalle de las relaciones junto a los ficheros/clases involucradas.

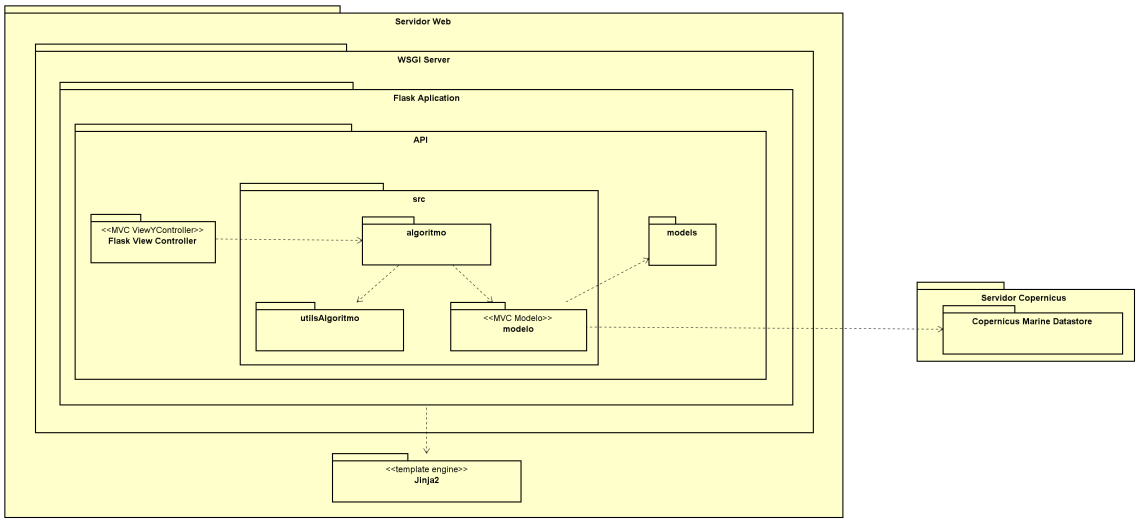

Figura 5.2: Arquitectura del servidor.

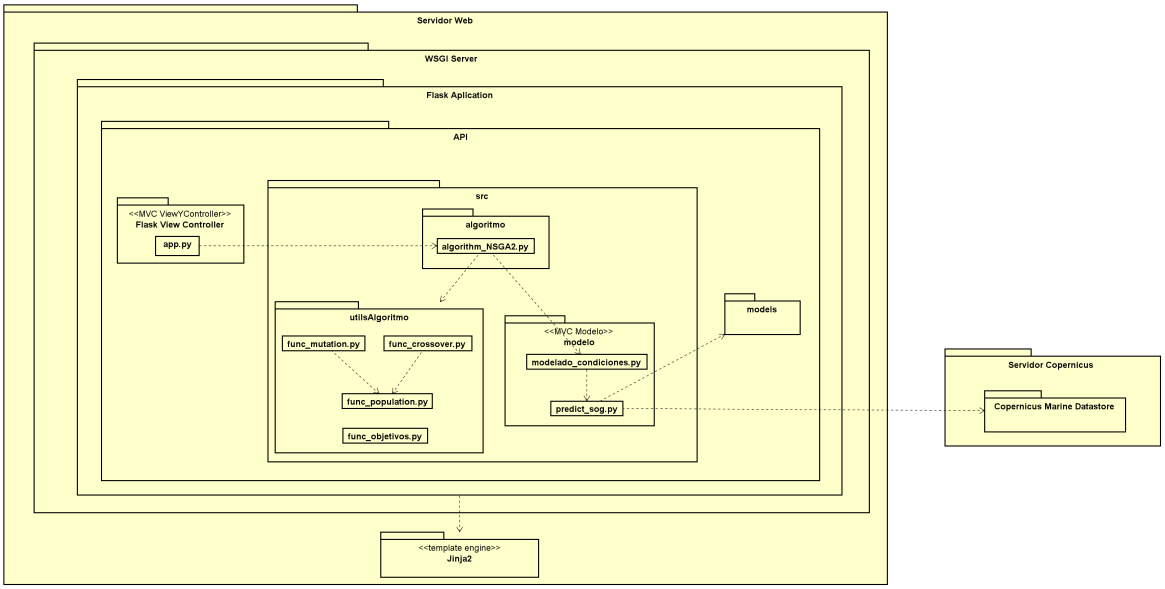

Figura 5.3: Arquitectura detallada del servidor.

5.5.1. Arquitectura Flask

Flask es un framework minimalista y flexible que permite crear aplicaciones web basado en dos componentes fundamentales: Werkzeug y Jinja2. Werkzeug es una biblioteca que proporciona utilidades para manejar las peticiones y respuestas HTTP, así como otras funcionalidades como el enrutamiento o el manejo de errores. Jinja2 es el motor de plantillas que usa Flask para generar el contenido dinámico de las respuestas, permitiendo combinar código HTML junto con expresiones, variables e instrucciones de Python.

Como se comentaba anteriormente, Flask es una herramienta para crear aplicaciones web, pero no es un servidor web. Un servidor web es el encargado de recibir y procesar las peticiones HTTP de los clientes y enviar las respuestas generadas por la aplicación web. Flask puede usar diferentes servidores, el integrado o uno externo como los populares Gunicorn o uWSGI. Sin embargo, la aplicación Flask no se comunica directamente con el servidor web, sino que utiliza una interfaz llamada (Web Server Gateway

Interface). WSGI es una especificación que establece la forma en que el servidor web y la aplicación web en Python se comunican, utilizando un protocolo estándar basado en la invocación de objetos (callable objects) que reciben las solicitudes y devuelven las respuestas. Como implementación por defecto, Flask utiliza Werkzeug de WSGI, que es una biblioteca subyacente de Flask que proporciona funcionalidades a la hora de manejar las peticiones y respuesta HTTP, enrutamiento o manejo de errores [8].

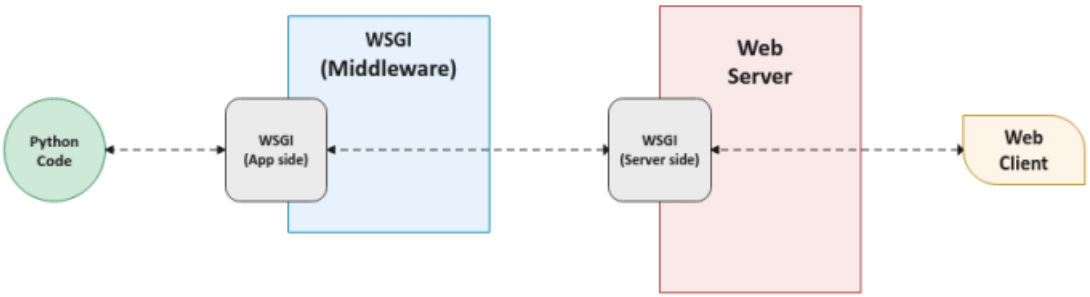

Figura 5.4: Componentes arquitectura WSGI a alto nivel. Fuente:[7]

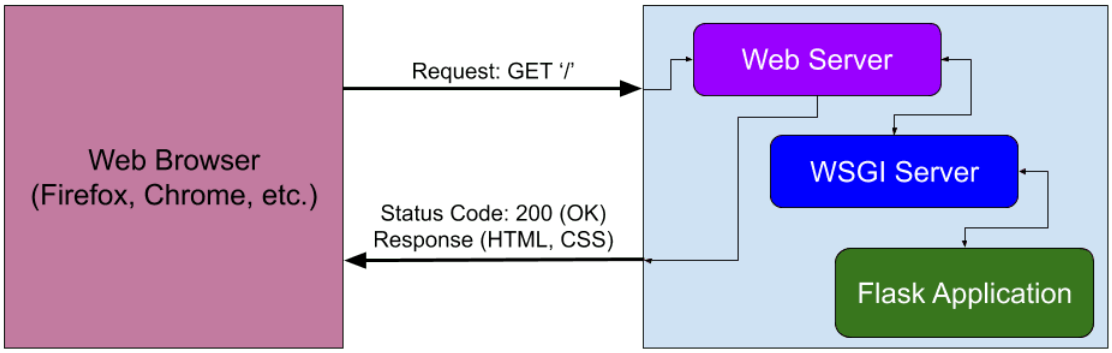

Figura 5.5: Procesado de la solicitud Flask a alto nivel. Fuente:[8]

La aplicación Flask, se refiere a una instancia de la clase Flask que representa la aplicación web en sí. Esta se encarga de registrar las funciones que maneja las solicitudes entrantes. Estas funciones, comúnmente conocidas como vistas o controladores, definen la lógica de cómo se debe responder a una solicitud HTTP específica. Al registrar estas funciones con la aplicación, Flask sabe qué vista se debe ejecutar cuando llega una solicitud a una URL determinada, y además, puede acceder a los datos enviados por el cliente a través de la petición (parámetros de la URL, datos del formulario, ...).

Las respuestas generadas por las vistas pueden contener contenido estático o dinámico. La diferencia entre ambos es que el primero no cambia según la petición o el contexto; mientras que el segundo se genera a partir de la información del contexto o petición. Para generar este contenido dinámico, Flask utiliza Jinja2. Las plantillas Jinja2 se almacenan en archivos con extensión .html y se pueden renderizar desde las vistas pasando las variables que se quieren usar en la plantilla.

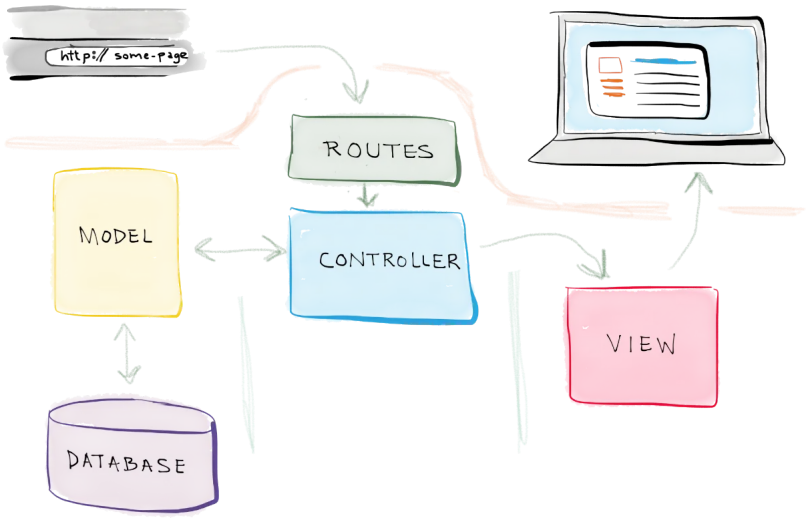

Figura 5.6: Descripción enrutamiento Flask en un patrón software MVC. Fuente:[9]

5.5.2. Diseño detallado del servidor

En este apartado se muestran las funciones específicas de cada clase/fichero del lado del servidor y los atributos necesarios en cada uno. En la que en el paquete *Flask View Controller* se encuentra el archivo instanciador de la clase Flask junto a la definición de las rutas de la API y su control. Posteriormente en *src*, se encuentran los paquetes de *algoritmo* encargado de toda la lógica de preparar los datos meteorológicos para ejecutar el algoritmo genético para obtener las rutas óptimas. Para poder ejecutar los pasos del algoritmo genético se hace uso de las funciones contenidas en los ficheros del paquete *utilsAlgoritmos* que encapsulan la lógica de las fases del algoritmo genético. Y para poder gestionar las peticiones y modelado de los datos marinos se encargan las funciones de los ficheros del paquete *modelo*.

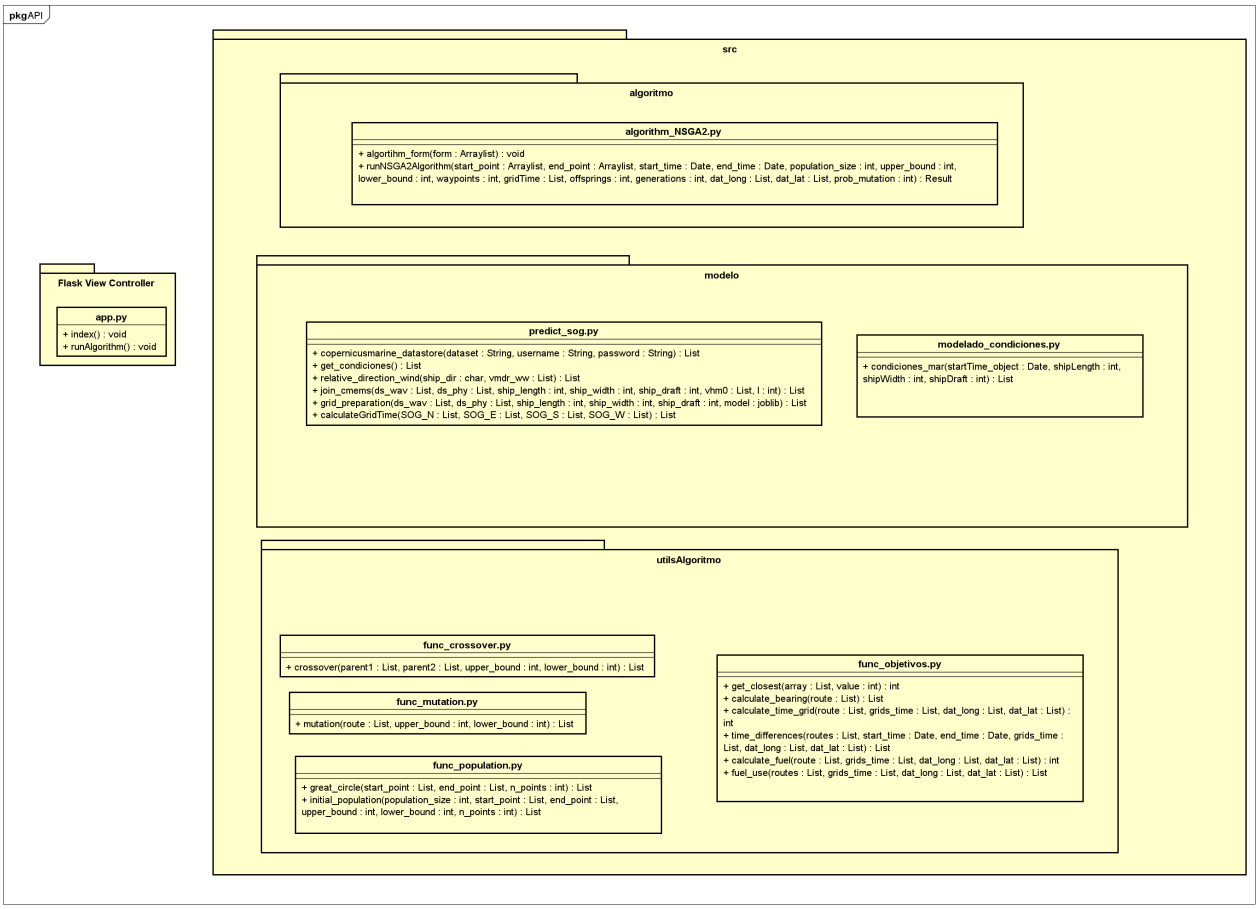

Figura 5.7: Diseño detallado por clases del servidor.

5.5.3. Arquitectura algoritmo

En esta Sección se representa el diseño de la parte algorítmica y de las clases implicadas. En él podemos apreciar la utilización del Patrón Estrategia definido con anterioridad. El encapsulamiento del algoritmo viene dado por la librería Pymoo, lo que permite ocultar los detalles internos de la implementación completa del algoritmo. La llamada de ejecución viene dada por la clase que representa el fichero *algorithmNSGA2.py*, que es la encargada de crear e integrar todo el conjunto de clases entorno al framework *Pymoo* necesarias para definir el entorno y condiciones del problema de optimización de rutas.

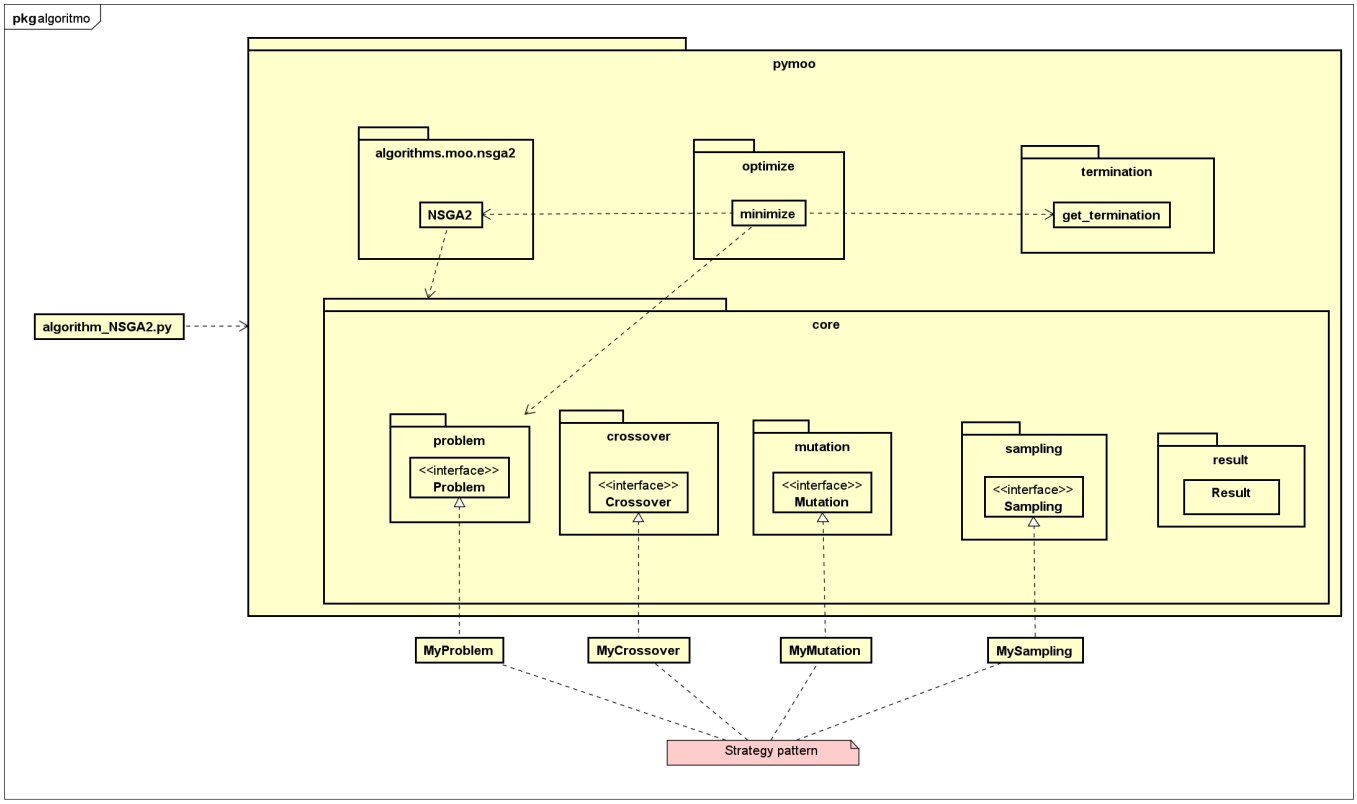

Figura 5.8: Arquitectura detallada del algoritmo.

5.5.4. Diseño detallado del algoritmo

En esta Sección se muestran las diferentes funciones específicas de cada clase que gestionan el flujo del algoritmo NSGA-II. Las más destacadas son aquellas dentro de las clases del patrón Estrategia que sobreescriben los métodos heredados de la clases pertenecientes al paquete *core* definidas por la biblioteca *Pymoo*.

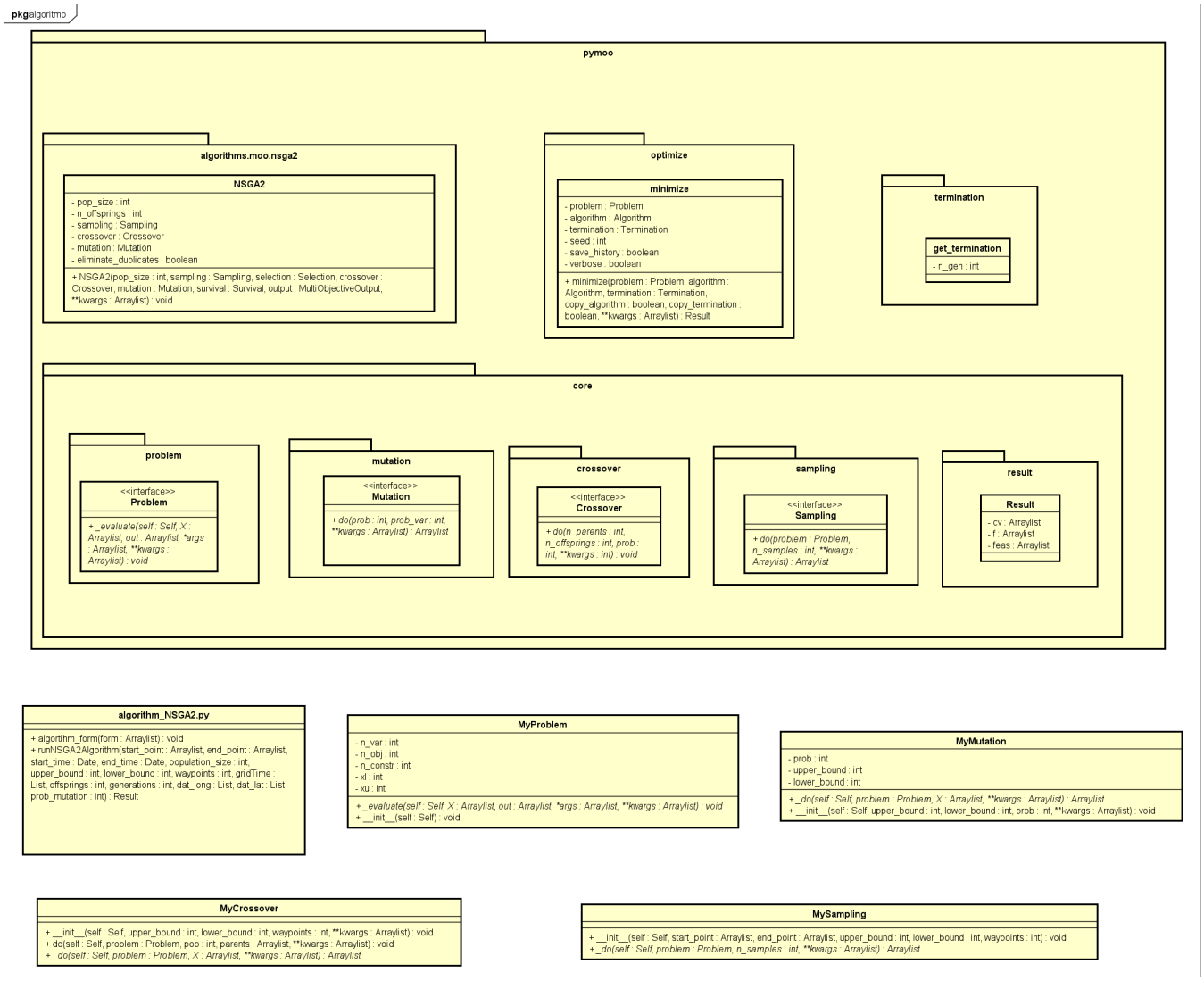

Figura 5.9: Diseño detallado por clases del algoritmo.

5.6. Arquitectura del cliente

Para la arquitectura del Cliente, hemos optado de nuevo por el patrón MVC. En el paquete templates contiene los ficheros .html de plantilla que utiliza Jinja2 para mostrar los datos del modelo y los datos que introduce el usuario. A su vez contará con funciones JavaScript que define las funciones que se deben ejecutar al interactuar con la vista, y permiten realizar peticiones asíncronas al servidor para obtener los datos de las rutas marítimas óptimas, convertidos a formato json, y posteriormente junto a la librería *Plotly.js* representarla en la vista.

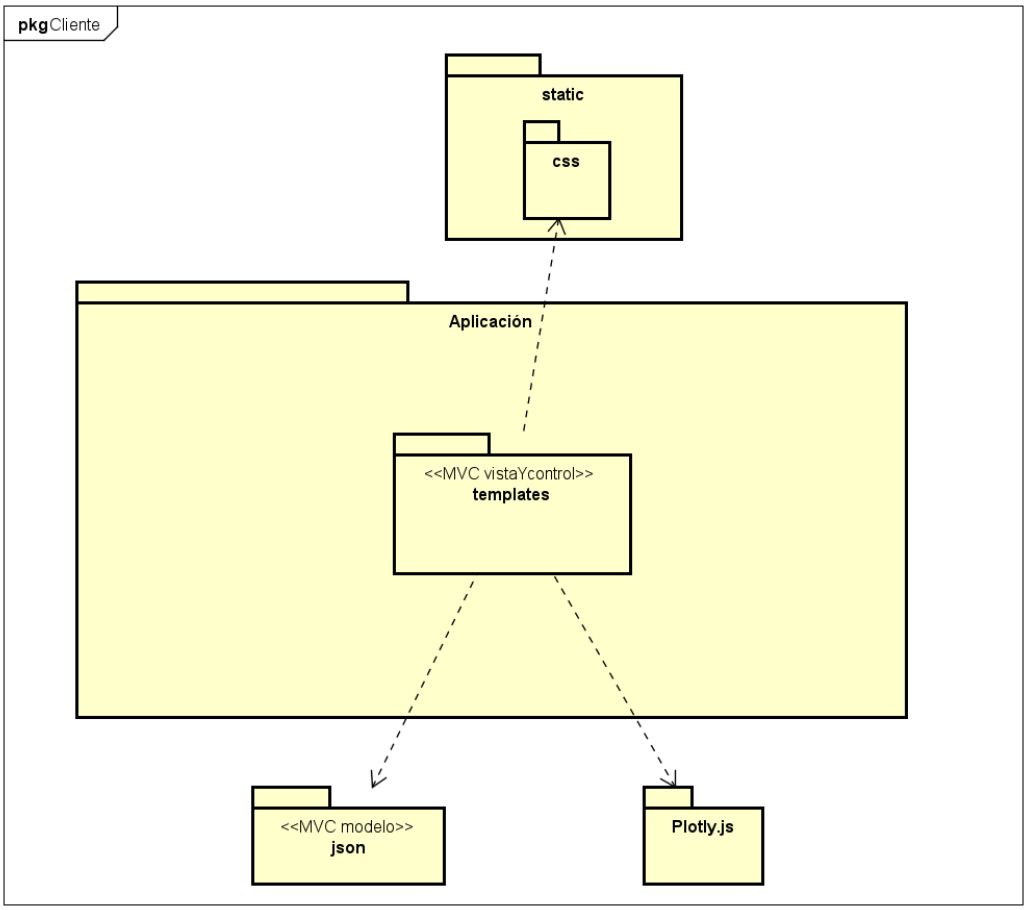

Figura 5.10: Arquitectura del cliente.

5.7. Interfaz de Usuario

A la hora del desarrollo de la interfaz de la página web se quiso dar importancia a la representación de la ruta marítima sobre el resto de la web. Para ello hacemos uso de un diseño minimalista y sencillo y el patrón de enfoque único, basados en el patrón de la **eficiencia**.

El diseño minimalista se caracteriza por la utilización de elementos visuales de interacción mínimos, con el objetivo de eliminar cualquier distracción. En nuestra página web, esto implica que la zona del mapa, donde se representaran las rutas óptimas y las condiciones marítimas, destacan como elementos principales.

Además esto hila con el patrón de enfoque único, lo que significa que se presenta al usuario una única tarea u objetivo: introducir los datos del barco y de la ruta que se va a realizar y ejecutar el algoritmo para obtener las rutas óptimas. Esto evita que el usuario se sobrecargue con demasiada información o funcionalidades adicionales, manteniendo el enfoque en el objetivo principal.

Otros factores a tener en cuenta, que siguen relacionados con el tema de la eficiencia, podrían ser el feedback y la visualización del mapa. En torno a ellos se ha prestado atención en proporcionar al usuario información sobre el estado del sistema y retroalimentación del proceso, para que el usuario no se frustre pensando que la página está inactiva o congelada. Además la visualización y los datos que informen acerca de la ruta juega un papel relevante ya que que mediante la librería *Plotly.js* nos permite representarlo mediante un mapa interactivo y que el usuario pueda hacer zoom, desplazarse por la ruta

y otras interacciones, además de resaltar puntos importantes de la ruta o dar contexto.

5.7.1. Mockups diseño interfaz

En la figura se muestra el mockup final de la aplicación. Los realizados en fases anteriores no se incluyen.

En la Fig. 5.11 se muestra la pantalla inicial de la aplicación, en la que se puede apreciar el formulario lateral mediante el cual el usuario introduce los datos del barco, de la ruta y los parámetros del algoritmo, si modifica alguno. Una vez el usuario da al botón de ejecutar algoritmo, y si los valores son correctos, se manda una petición al algoritmo para que calcule las rutas óptimas y devuelva aquella con el menor tiempo de navegación y aquella cuyo coste de fuel sea menor. La representación de estas rutas, junto a los datos meteorológicos, dio lugar a la realización del segundo mockup, Fig. 5.12.

Figura 5.11: Mockup Pantalla Inicial.

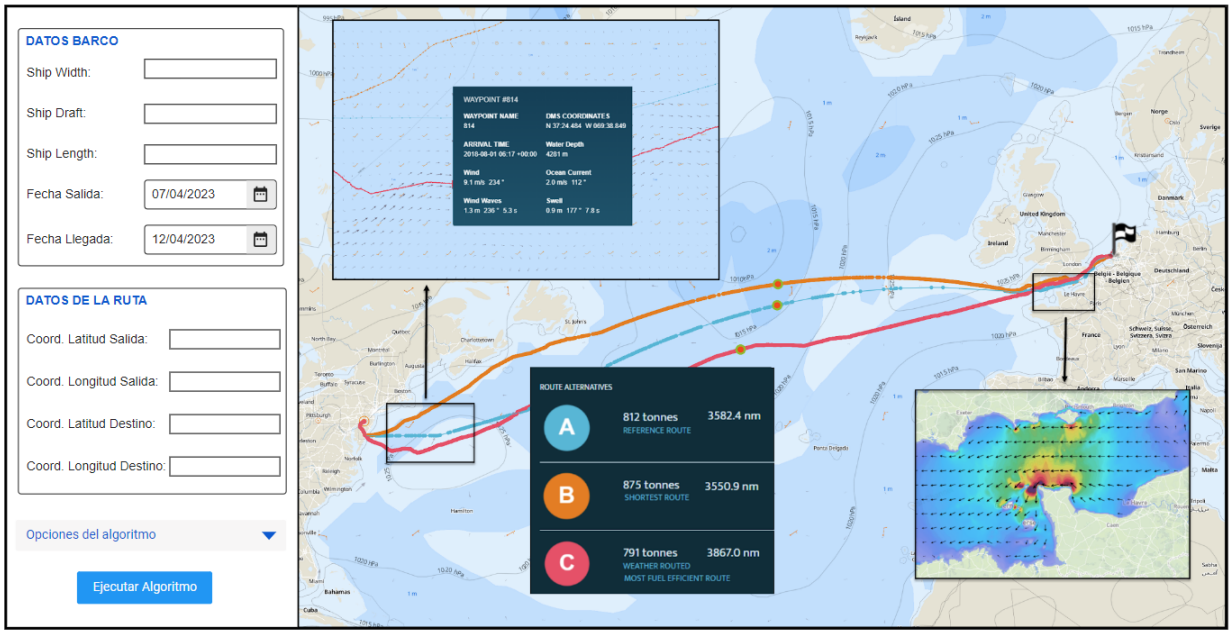

Figura 5.12: Mockup Pantalla Ejecución Algoritmo (Imagen del mapa de la derecha utilizada obtenida del software NAPA).[1]

5.8. Privacidad de los datos

Este factor al no contar con una base de datos para almacenar la información del cliente no ha necesitado tratamiento. Nuestro proyecto no tiene ningún tipo de gestión privada de los datos, al no almacenar ninguno; y los únicos datos con los que se trata son las condiciones marítimas que se proporcionan del servidor de Copernicus, cuyo acceso es de forma gratuita.

[1] https://www.napa.fi/making-weather-routed-voyage-planner-openly-available/

Capítulo 6

Modelo de Datos

6.1. Introducción

En la Sección 3.5.1 se comentó el fundamento teórico y matemático a la hora de modelizar la velocidad sobre el suelo (SOG). En él se introducía como, en este Trabajo de Fin de Grado, se había optado por seguir la técnica planteada en M. Abebe et al., 2020 [10].

En este Trabajo Fin de Grado, nuestro objetivo es aplicar un modelo de aprendizaje automático similar para predecir el SOG utilizando datos de Copernicus Marine Data Store, que proporciona acceso libre y gratuito a productos de datos oceánicos, incluidos modelos de olas y física marina, así como datos AIS. Se explorarán algunos de los modelos propuestos con mejores resultados, comparando su precisión y tiempo de cálculo.

Con ello ya somos capaces de, teniendo en cuentas las condiciones marítimas, predecir la velocidad en los puntos que atraviesa la ruta y junto a la distancia recorrida a dicha velocidad ser capaces de modelar nuestro objetivo de minimizar el tiempo de ruta. Este aspecto, junto al resto de metodología llevada a cabo, es discutida en el Capítulo 7.

6.2. Descripción de los datos

6.2.1. Copernicus Marine Data Store

El Copernicus Marine Data Store es una plataforma en línea gratuita y abierta que proporciona acceso a una amplia gama de productos y servicios de datos oceánicos. Estos productos y servicios se basan en observaciones satelitales y "in situ", modelos numéricos y análisis expertos del estado físico, biogeoquímico y del hielo marino del océano global y los mares regionales europeos. El objetivo principal del Copernicus Marine Data Store es apoyar la implementación de la política marina y mejorar la comprensión del medio ambiente marino y su impacto en el clima y la sociedad. Además, la plataforma fomenta la innovación científica al proporcionar un catálogo robusto de datos oceánicos, un informe anual sobre el estado del océano, indicadores de monitoreo oceánico y herramientas interactivas de visualización oceánica. El Copernicus Marine Data Store forma parte del Servicio Marino Copernicus, que es el componente marino del Programa de Observación de la Tierra de la Unión Europea Copernicus [108].

El Servicio Marino Copernicus se basa en una red de proveedores de datos e información que incluye centros nacionales e internacionales de oceanografía operacional, agencias espaciales, institutos de investigación y organizaciones no gubernamentales. El Servicio Marino Copernicus ofrece una variedad de

productos y servicios para diferentes usuarios y sectores, como la seguridad marítima, la pesca sostenible, la protección del medio ambiente marino, el cambio climático, la energía renovable oceánica, el turismo costero y la educación oceánica.

El Copernicus Marine Data Store permite a los usuarios descargar o visualizar datos que incluyen retropronósticos (*hindcasts*), pronósticos actuales (*nowcasts*) y pronósticos futuros (*forecasts*) de diferentes variables oceánicas, como la temperatura, la salinidad, la altura de la superficie del mar, el hielo marino, el oxígeno, los nutrientes, el plancton y las olas. Los datos se organizan en diferentes áreas geográficas, como el océano global, el océano Antártico, el océano Ártico, el Atlántico Norte, el Mar Báltico, el Mar Negro y el Mar Mediterráneo. Los datos también se clasifican según el tipo de característica (grilla o vector), la resolución temporal (instantánea o media), el nivel de procesamiento (L3 o L4) y el centro de origen.

6.2.1.1. Global Ocean Physics Analysis and Forecast

El dataset Global Ocean 1/12 grado Physics Analysis and Forecast updated Daily es un producto del servicio marino de Copernicus que proporciona un análisis y pronóstico diario de las condiciones físicas del océano global a una resolución espacial de 1/12 grado (unos 8 km). El dataset es generado por el sistema PSY4V3R1, que es el nombre del sistema operacional de Mercator Ocean que produce este producto. Este sistema utiliza el modelo oceánico NEMO (*Nucleus for European Modelling of the Ocean*), que es una plataforma numérica para simular la circulación oceánica y sus interacciones con la atmósfera, el hielo marino y los ciclos biogeoquímicos. El sistema utiliza un esquema de asimilación de datos variacional 4D (4D-VAR) que combina las observaciones satelitales y en situ con el modelo mediante un algoritmo de minimización. El dataset incluye campos horarios de temperatura, salinidad, corrientes y nivel del mar en la superficie, y campos diarios de los mismos parámetros en 50 niveles verticales desde la superficie hasta el fondo del océano. También se incluye un campo especial de corriente superficial que incluye el efecto de las olas y las mareas llamado SMOC (*Surface merged Ocean Current*). El dataset cubre una ventana temporal deslizante de dos años completos y se actualiza diariamente con un pronóstico de 10 días. Este dataset es útil para estudiar la variabilidad y el cambio climático del océano, así como para apoyar aplicaciones operativas como la navegación, la pesca, la gestión costera y la seguridad marítima [11].

En la siguiente tabla se describen las variables utilizadas del dataset:

Dataset : cmems_mod_glo_phy_anfc_0.083deg_P1D-m		
Variable	**Descripción**	**Unidad**
zos	Altura de la superficie del mar	m
mlotst	Espesor de la capa mezclada del océano	m
tob (bottomT)	Temperatura del fondo marino	℃
siconc	Concentración de hielo	l
sithick	Espesor del hielo marino	m
usi	Velocidad hacia el este del hielo marino	m/s
vsi	Velocidad hacia el norte del hielo marino	m/s
Dataset : cmems_mod_glo_phy-so_anfc_0.083deg_P1D-m		
Variable	**Descripción**	**Unidad**
so	Salinidad del agua de mar	psu
Dataset : cmems_mod_glo_phy-thetao_anfc_0.083deg_P1D-m		
Variable	**Descripción**	**Unidad**
thetao	Temperatura potencial del agua de mar	℃
Dataset : cmems_mod_glo_phy-cur_anfc_0.083deg_P1D-m		
Variable	**Descripción**	**Unidad**
uo	Velocidad de la corriente oceánica hacia el este	m/s
vo	Velocidad de la corriente oceánica hacia el norte	m/s

Tabla 6.1: Descripción del dataset y variables utilizadas.

6.2.1.2. Global Ocean Waves Analysis and Forecast

Este producto se basa en el sistema operativo global de análisis y pronóstico de olas del océano de Météo-France con una resolución de 1/12 grado, que utiliza el modelo de olas MFWAM, un modelo de tercera generación que incorpora los últimos avances en la física y la parametrización de las olas. El modelo MFWAM se alimenta de vientos analizados y pronosticados cada 6 y 3 horas respectivamente por el sistema atmosférico IFS-ECMWF. El producto incluye campos instantáneos cada 3 horas de parámetros integrados de las olas a partir del espectro total (altura significativa, período, dirección, deriva de Stokes, etc.), así como las siguientes particiones: la ola del viento, la ola primaria y secundaria del oleaje. El producto también ofrece un conjunto especial de datos para la corriente superficial que incluye también la deriva por oleaje y marea llamada SMOC (Surface Merged Ocean Current). El *Global Ocean Waves* proporciona análisis 4 veces al día y un pronóstico de 10 días a las 0:00 UTC. El modelo de onda MFWAM utiliza la partición para dividir el espectro de oleaje en olas de marea primarias y secundarias [109].

En la siguiente tabla se describen las variables utilizadas del dataset:

Dataset : cmems_mod_glo_wav_anfc_0.083deg_PT3H-i		
Variable	**Descripción**	**Unidad**
VHM0	Altura significativa de las olas en la superficie del mar	m
VTM10	Período promedio de las olas en la superficie del mar a partir de la densidad espectral de varianza inversa de frecuencia	s
VTM02	Período promedio de las olas en la superficie del mar a partir de la densidad espectral de varianza de la segunda frecuencia	s
VTPK	Período de las olas en el pico de la densidad espectral de varianza	s
VMDR	Dirección media de las olas en la superficie del mar	grados decimales
VPED	Dirección principal de las olas en el pico de la densidad espectral de varianza	grados decimales
VSDX	Velocidad de desplazamiento de Stokes en la dirección x	m/s
VSDY	Velocidad de desplazamiento de Stokes en la dirección y	m/s
VHM0_WW	Altura significativa de las olas de viento en la superficie del mar	m
VTM01_WW	Período de onda de viento promedio de momentos espectrales (0,1)	s
VMDR_WW	Dirección media de las olas de viento en la superficie del mar	grados decimales
VHM0_SW1	Altura significativa de la oleada primaria principal en la superficie del mar	m
VTM01_SW1	Periodo promedio de ondas de oleaje primario principal en la superficie del mar	s
VMDR_SW1	Dirección media de las olas de oleaje primario principal en la superficie del mar	grados decimales
VHM0_SW2	Altura significativa de la oleada secundaria principal en la superficie del mar	m
VTM01_SW2	Periodo promedio de ondas de oleaje secundaria principal en la superficie del mar	s
VMDR_SW2	Dirección media de las olas de oleaje secundaria principal en la superficie del mar	grados decimales

Tabla 6.2: Descripción del dataset y variables utilizadas.

6.2.2. Automatic Identification System (AIS)

El Sistema de Identificación Automática (AIS, por sus siglas en inglés) es un sistema automático de seguimiento de embarcaciones que se introdujo para promover una navegación marítima segura y eficiente (Organización Marítima Internacional 2001 [110], Harati-Mokhtari et al. 2007 [111]). Uno de los objetivos era mejorar la calidad de la vigilancia del tráfico marítimo. El AIS proporciona información sobre la identidad del buque, su ubicación, velocidad, rumbo, datos del buque y el viaje que está realizando. Esta información se transmite a través de canales de radio VHF entre sistemas AIS como estaciones base, boyas, buques y centros VTS (Servicios de Tráfico de Buques).

Se considera que los datos del AIS son fiables, ya que los transmiten tanto el buque como las estaciones costeras. Sin embargo, hay algunos factores que pueden afectar a la precisión de los datos AIS, como la calidad del equipo AIS del buque, la distancia entre el buque y la estación costera, las condiciones meteorológicas y la presencia de otros buques en la zona.

Aunque el envío de datos AIS está automatizado, sólo la información dinámica se lee directamente de los sensores de navegación del buque. Los datos estáticos y de la travesía son introducidos manualmente por la tripulación del barco, lo que puede afectar a la validez de los datos. Los datos dinámicos también pueden contener errores debidos a problemas de integración con los sensores del barco.

Harati-Mokhtari et al. (2007) [111] señalan que los datos estáticos y de la travesía en muchos casos son erróneos. Informaron de problemas como que el 56 % de los tipos de buque eran erróneos y que las dimensiones eran incorrectas en el 47 % de los mensajes. Además, los campos de destino y ETA (Tiempo Estimado de Llegada) no se actualizaban con frecuencia o contenían información errónea debido a errores de introducción de los datos.

A pesar de estas limitaciones, el AIS es una herramienta valiosa para mejorar la seguridad y la eficacia de la navegación marítima. Proporciona información precisa, oportuna y completa sobre los buques, que puede utilizarse para diversas aplicaciones, como la vigilancia del tráfico de buques, la gestión portuaria, la búsqueda y salvamento, la seguridad marítima y la vigilancia medioambiental.

Vigilancia del tráfico marítimo

Los datos AIS pueden utilizarse para seguir el movimiento de los buques e identificar posibles colisiones. Esta información puede ser utilizada por los Servicios de Tráfico de Buques (VTS) para gestionar el tráfico en vías navegables muy transitadas y garantizar la seguridad de los buques y las personas.

Gestión portuaria

En la gestión portuaria, la información proporcionada por los sistemas AIS resulta de gran utilidad para planificar operaciones, dirigir los barcos a los muelles y supervisar el movimiento de los mismos dentro del puerto, asegurando la seguridad de todos los implicados.

Búsqueda y salvamento

En situaciones de búsqueda y rescate, los sistemas AIS son vitales para localizar barcos desaparecidos y coordinar operaciones de salvamento. La información que proporcionan permite hacer un seguimiento preciso del movimiento de los barcos, identificar los que puedan estar en peligro y dirigir a los equipos de rescate al lugar del incidente.

Seguridad marítima

Además de la vigilancia del tráfico marítimo, los datos del AIS también pueden emplearse para la seguridad marítima, mediante el seguimiento de los barcos y la identificación de posibles amenazas. Esta información permite hacer un seguimiento de los movimientos de los barcos, identificar aquellos que pueden ser de interés para los organismos de seguridad y dirigir a las fuerzas de seguridad al lugar de los hechos.

Vigilancia medioambiental

La información que proporcionan los sistemas AIS también puede ser utilizada para el monitoreo ambiental, permitiendo el seguimiento de la ubicación de los barcos y la identificación de posibles riesgos ambientales. Esto puede ayudar a rastrear los movimientos de los barcos, identificar aquellos que estén vertiendo contaminantes y dirigir a las agencias ambientales al lugar del incidente.

Dataset : AIS DATA NOAA			
Variable	**Descripción**	**Ejemplo**	**Unidad**
MMSI	Servicio móvil marítimo. Valor de identidad	477220100	
BaseDateTime	Fecha y hora UTC completas	2017-02-01T20:05:07	
LAT	Latitud	42.35137	grados decimales
LON	Longitud	-71.04182	grados decimales
SOG	Velocidad sobre el suelo	5.9	nudos
COG	Curso sobre tierra	47.5	grados
Heading	Ángulo de rumbo real	45.1	grados
VesselName	Nombre que figura en la licencia de radio de la estación	OOCL Malaysia	
IMO	Número Marítimo Internacional Organización Marítima Internacional	IMO9627980	
CallSign	Indicativo de llamada asignado por FCC	VRME7	
VesselType	Tipo de buque según se define en especificaciones NAIS	70	
Status	Estado de navegación definido por el COLREGS	3	
Length	Longitud del buque	71.0	metros
Width	Ancho del buque	12.0	metros
Draft	Calado	3.5	metros
Cargo	Tipo de cargo	70	
TransceiverClass	Clase de transceptor AIS	A	

Tabla 6.3: Descripción del dataset AIS.

6.3. Modelización

Inicialmente, deben recopilarse y modelarse datos que tengan información relevante con los objetivos para poder crear una base para el cálculo posterior de las rutas. Para ello se utilizarán los datos del AIS para la información básica de los barcos, juntos con los datos meteorológicos marinos.

Para encontrar la ruta adecuada, es necesario predecir con exactitud la velocidad del buque. Estudios anteriores demostraron que la velocidad del buque puede estimarse evaluando la pérdida de velocidad del buque en función de su resistencia mediante análisis teóricos. Sin embargo, el uso de ecuaciones teóricas no puede aplicarse a la mayoría de los buques en diversas condiciones de funcionamiento.

Para superar este problema, replicaremos el modelo basado en datos propuesto en M. Abebe et al., 2020 [10]. En este artículo se propone un marco de análisis de datos marítimos basado en datos AIS y meteorológicos marinos para predecir la velocidad del buque sobre el fondo (SOG), lo que determina la ruta marítima más económica que puede reducir los gastos de combustible. Se utilizaron varias técnicas de regresión para generar modelos y se comparó su precisión y eficiencia con datos reales.

Las secciones restantes de aplicación de dicho modelo se organizan de la siguiente forma: una Sección 6.3.1 en la que se analizan las técnicas a la hora de la extracción de datos, Sección 6.3.2 en la que se detallan los pasos para la transformación de los datos, en la Sección 6.3.3 a partir de los datos se hace un estudio de correlación de las variables y finalmente en la Sección 6.3.4 se describe cómo se han llevado a cabo estos modelos predictivos, los resultados arrojados y la discusión de dichos resultados.

Es relevante mencionar que todos los procesos de adquisición, fusión, preprocesado y evaluación de los modelos implementados fueron ejecutados en una máquina virtual suministrada por el Departamento de Informática. Esta máquina cuenta con 4 núcleos, 32GB de RAM y 128GB de tamaño de disco; ya que el tiempo de estos procesos era elevado y requería unas ciertas especificaciones de la máquina.

En la Fig. 6.1 se presenta una representación esquemática de la metodología propuesta.

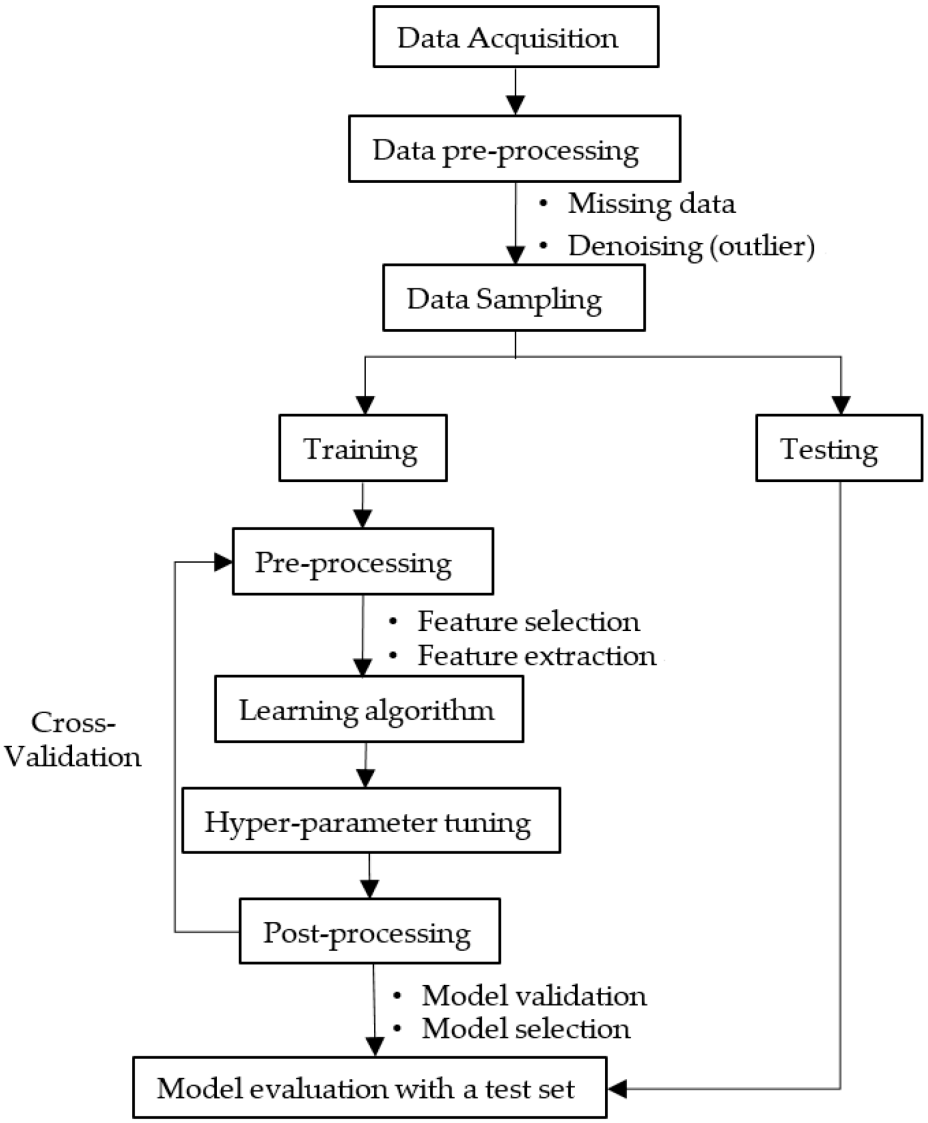

Figura 6.1: Esquema de la metodología propuesta. Fuente: [10]

6.3.1. Extracción de los Datos

En el ámbito del aprendizaje automático, los datos desempeñan un papel crucial en el proceso de entrenamiento y validación del modelo. Por lo tanto, es fundamental manejarlos correctamente con el fin de mejorar la eficacia global del modelo.

Para obtener información básica sobre los barcos, se utiliza el llamado sistema de identificación automática en barcos (AIS). Además de los datos básicos de los barcos, se requiere información ambiental para modelar la influencia del clima en la navegación. Para este propósito, se utilizan dos conjuntos de datos del CMEMS (Servicio de Vigilancia Marina Copernicus). En primer lugar, datos de oleaje que

incluyen información sobre altura, período, pico y dirección de las olas; y en segundo lugar, datos físicos que contienen información sobre factores ambientales como temperatura, salinidad, velocidad y hielo.

Todos los conjuntos de datos se filtraron para seleccionar los parámetros relevantes para nuestra optimización. Además, los datos del AIS se filtraron en función del contenido, ya que decidimos enfocar nuestro modelo en un tipo de barco específico. Con este fin, los datos se filtraron específicamente para los barcos de carga en movimiento, con una velocidad superior a 7 nudos y cuyo tamaño se ajustara al ancho del Canal de Panamá.

Con el fin de obtener los datos más actualizados y en función del alcance de este trabajo, los datos de los tres conjuntos diferentes se fusionaron para el año 2022 en las zonas UTM (sistema de coordenadas universal transversal de Mercator) 1-20 [1], lo que resultó en un número de 192.435 puntos de datos. Para la fusión, se tomó la marca de tiempo de los datos del AIS como base y se adjuntaron los datos del paso de tiempo anterior de los otros dos conjuntos de datos.

A la hora de la adquisición de estos datos se ha desarrollado eN un notebook [2], que podemos describir en las siguientes partes: En primer lugar, se encarga de la descarga de los datos del AIS provenientes de la Oficina de Gestión Costera de la NOAA(Oficina Nacional de Administración Oceánica y Atmosférica del Departamento de Comercio de los Estados Unidos) [3]. Algunos fragmentos utilizan código producido por 52North [4]. Este código fue modificado para mejorar su eficiencia computacional y poder así descargar mayor cantidad de datos sin requerir tantos recursos.

Posteriormente se descargan los datos físicos y de ondas CMEMS correspondientes a los puntos de datos recibidos de los datos AIS. Para que en tercer lugar todas las partes de los datos AIS, de las ondas CMEMS y de la física CMEMS que nos interesan se fusionan en un único archivo Dataframe/CSV. Para ello, de nuevo se utilizó fragmentos de código de 52North [5]. Debido a la incorporación por nuestra parte de los datos físicos marítimos y las modificaciones en los accesos a los datos realizadas por Copernicus, se han realizado una serie de cambios para adaptarlo a estas.

En estos notebooks se utilizan diferentes librerías Python y tecnologías que se explicarán en la Sección 8.1

6.3.2. Preprocesado de los Datos

Los datos en su forma original, conocidos como "raw data", son los datos obtenidos en una etapa inicial y se someten a diversas modificaciones y adaptaciones para generar un conjunto de datos nuevo, mejorado y refinado. Este conjunto de datos procesado será el utilizado en el modelo de aprendizaje final.

Se ha seguido el preprocesado llevado a cabo en M. Abebe et al., 2020 [10] que consta de:

- La velocidad de navegación puede disminuir debido a diferentes resistencias del estado del mar; sin embargo, también existe la probabilidad de que sea reducida por el operador, especialmente alrededor del puerto al inicio y al final del viaje. Para reducir este tipo de errores de medición, en este estudio se descartaron los datos con menos de 7 nudos de SOG, considerados como maniobras.

[1] Más información sobre las zonas UTM: `https://es.wikipedia.org/wiki/Sistema_de_coordenadas_universal_transversal_de_Mercator` y `https://marinecadastre.gov/AIS/AIS%20Documents/UTMZoneMap2014.png`

[2] `https://gitlab.inf.uva.es/migchav/tfg-miguel-chaveinte/-/blob/master/notebooks/26_04_AIS_CMEMS_merge_data.ipynb`

[3] NOAA's Office for Coastal Management: `https://coast.noaa.gov/`

[4] 52North web: `https://52north.org/`. Código Github: `https://github.com/52North/MariGeoRoute/blob/main/data/ais/preprocessing_AIS.ipynb`

[5] Código Github:`https://github.com/52North/MariGeoRoute/blob/main/data/ais/merge_AIS_with_motu_rest_api.ipynb`

- Si el valor de los datos no está disponible (datos faltantes), se utiliza un valor atípico predeterminado para cada característica, como 102.2 para SOG, 511 para rumbo, 91 para latitud y 181 para longitud [10]. Estos valores se observaron en nuestros datos y se utilizaron para descartar los datos faltantes.

- El diagrama de dispersión de las características puede utilizarse para mostrar que los datos pueden contener ruido/valores atípicos debido a inconsistencias en la medición de los sensores o errores humanos, los cuales deben ser rechazados antes de entrenar los modelos. El puntaje Z (Z-score) es un método paramétrico para detectar valores atípicos en un espacio de características de diferentes dimensiones. Sin embargo, este método asume que los datos tienen una distribución gaussiana, lo que implica que los valores atípicos se distribuyen en las colas de la distribución, es decir, los puntos de datos están lejos del valor medio. Antes de establecer un umbral, que en este caso se denominó Z_{thr} , se normalizó el punto de datos x_i dado como Z_i utilizando la siguiente ecuación:

$$Z_i = \frac{x_i - \mu}{\sigma} \tag{6.1}$$

  donde μ y σ son la media y la desviación estándar de todos los x_i, respectivamente. Entonces, un valor atípico es aquel punto de datos cuyo valor absoluto es mayor o igual a Z_{thr} : $|Z_i| \geq Z_{thr}$.

  Por lo general, el valor umbral se establece en ± 3 [10]; sin embargo, los datos son no lineales y por lo tanto nos interesa eliminar aquellos casos extremos . Por lo tanto, se utilizó un valor umbral de ± 5 para todas las características, a fin de descartar valores que estuvieran extremadamente lejos del valor medio en ambas colas. La Fig. 6.2 y Fig. 6.3 muestran ejemplos de la distribución de datos de SOG, incluyendo datos normales y atípicos, que se detectaron utilizando el puntaje Z.

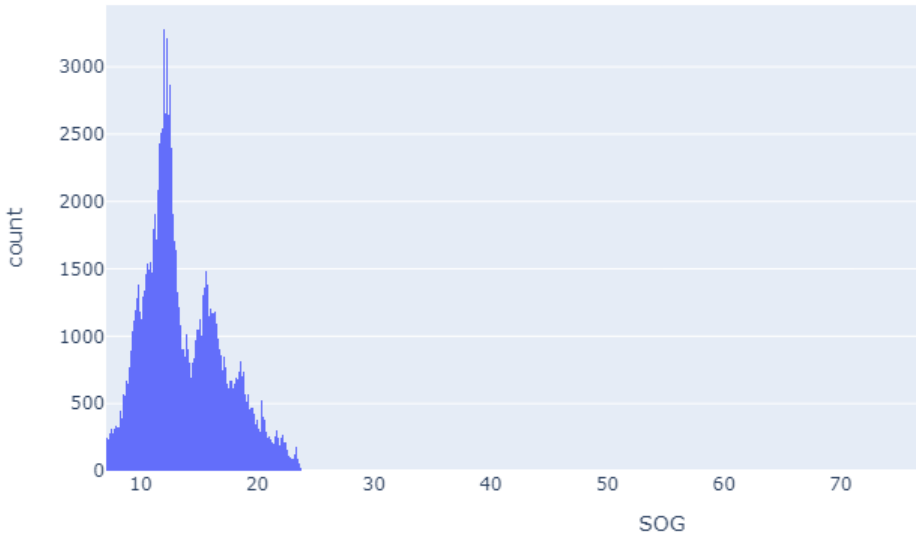

Figura 6.2: Distribución SOG

6.3.3. Extracción y selección características

Se realizó una selección de características para eliminar aquellas que no eran necesarias. Antes de realizar la selección de características, algunas de ellas se transformaron a un formato más conveniente. Las transformaciones realizadas en el conjunto de datos marítimos tienen como objetivo calcular la dirección relativa entre la dirección del barco y la dirección promedio del viento y las olas.

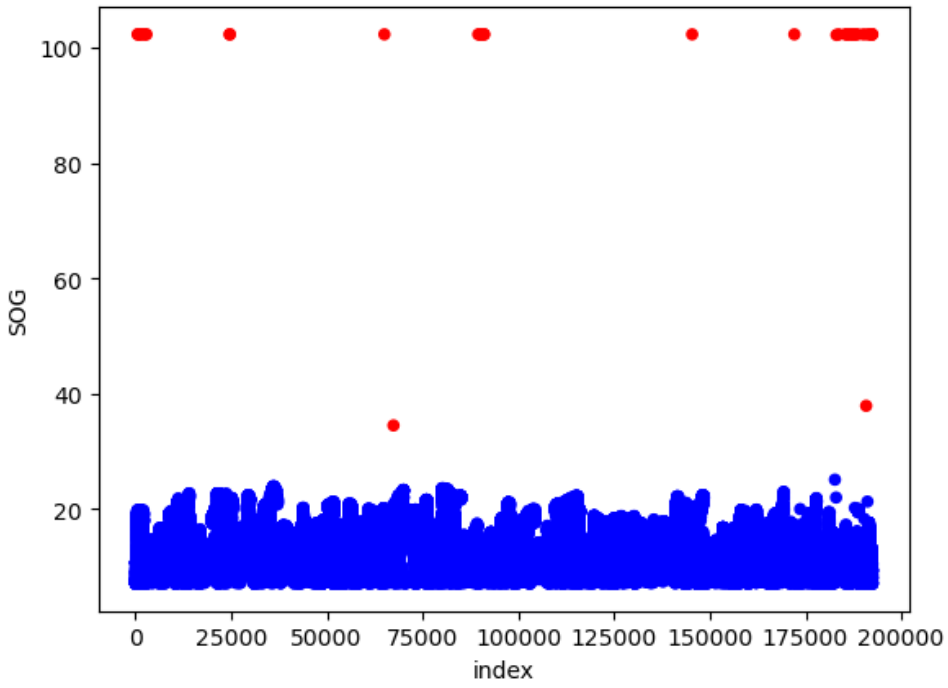

Figura 6.3: Diagrama de dispersión SOG. En rojo los outliers detectados por Z-score

- La columna "COG_true" se crea como una copia de la columna existente "COG", que representa la dirección del barco, ya que está altamente correlacionado con el rumbo verdadero.
- Si hay valores negativos en "COG_true", se les suma 360 para convertirlos en valores positivos y asegurarse de que estén en el rango correcto.
- La columna "rel_dir" calcula la diferencia absoluta entre "COG_true" y "VMDR_WW", lo que proporciona la diferencia angular entre la dirección del barco y la dirección promedio del viento y las olas.
- La columna "dir_4" se inicializa con el valor 2. Luego, se asignan diferentes valores basados en condiciones específicas:
  - Si la diferencia angular es menor a 45 grados o mayor a 315 grados, se asigna el valor 1 a "dir_4". Esto implica que la dirección del barco y la dirección promedio del viento y las olas están alineadas o casi alineadas.
  - Si la diferencia angular está entre 135 y 225 grados, se asigna el valor 3 a "dir_4". Esto indica que hay una diferencia significativa entre la dirección del barco y la dirección promedio del viento y las olas.

  Los números 1, 2 y 3 identifican la dirección relativa del viento respecto al barco. Por ejemplo, si el barco va al norte y el viento viene del este, el número sería 3 porque el viento viene de estribor. Si el barco va al sur y el viento viene del oeste, el número sería 1 porque el viento viene de proa. El número 2 significa que el viento viene de babor o de popa, dependiendo de la dirección del barco.
- Además, el cálculo de la resistencia del barco se realiza multiplicando las dimensiones del barco (longitud, anchura y calado) en la columna "resist".
- Por último se eliminaban aquellas filas que contenían algún valor faltante (Nan) en sus características.

Para eliminar características innecesarias, se realizó un filtro de alta correlación. Siguiendo el estudio propuesto en [10], la definición de filtro de alta correlación establece que si los valores observados de dos características de entrada son siempre iguales, significa que representan la misma entidad. Por lo tanto, las variables altamente correlacionadas se consideran como una sola variable. En la Fig. 6.4 se muestra el resultado de la matriz de correlación de 33 características de entrada; en las que los pares con una correlación mayor de 0,7 se tomaron como una única variables, reduciéndola a 21 variables. Las características finales seleccionadas se muestran en la Tabla **6.4**. La descripción estadística de estas variables se puede observar en el notebook del procesado [6].

[6] `https://gitlab.inf.uva.es/migchav/tfg-miguel-chaveinte/-/blob/master/notebooks/ConcatCSV.ipynb`

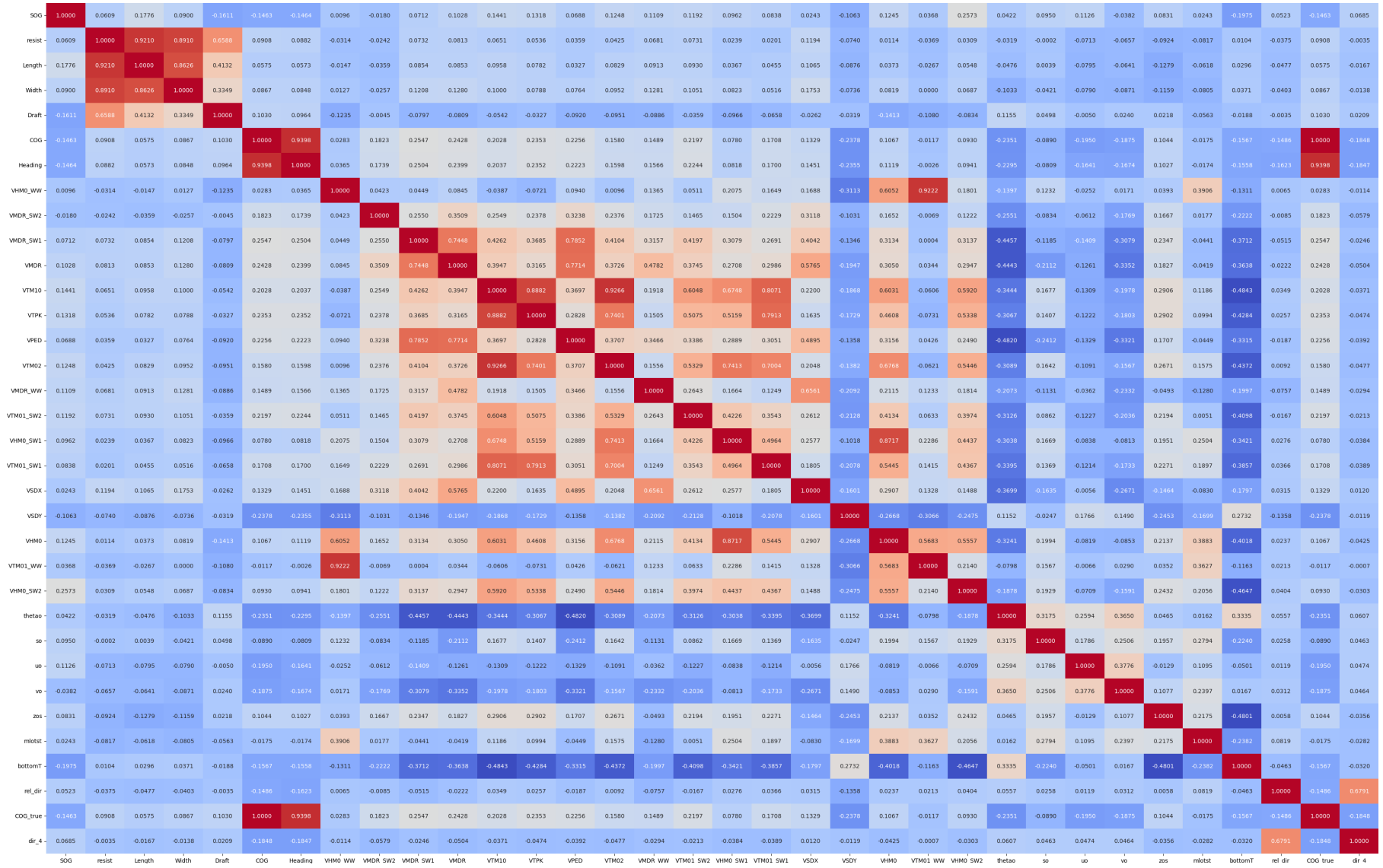

Figura 6.4: Matriz de correlación de características.

Observación	No.	Características	Unidades
Características de entrada	1	resist	m^3
	2	Draft (calado)	m
	3	VHM0_WW	m
	4	VMDR_SW2	grados
	5	VMDR_SW1	grados
	6	VTM10	s
	7	VMDR_WW	grados
	8	VTM01_SW2	s
	9	VHM0_SW1	m
	10	VSDX	m/s
	11	VSDY	m/s
	12	VHM0	m
	13	VHM0_SW2	m
	14	thetao	℃
	15	so	psu
	16	uo	m/s
	17	vo	m/s
	18	zos	m
	19	mlotst	m
	20	bottomT	℃
	21	dir_4	-
Variable objetivo	1	SOG	knots (nudos)

Tabla 6.4: Características finales seleccionadas.

La correlación entre SOG y los demás factores de entrada no es alta debido a que la velocidad del buque está determinada principalmente por el par motor y las RPM del buque; además, otras características relacionadas con el clima tienen una correlación relativamente baja. Si se hubieran recopilado datos sobre los motores de los buques, se podrían haber incluido características altamente correlacionadas como características principales. Sin embargo, solo se utilizan datos AIS y meteorológicos debido a que no se cuentan con dichos datos de motor por razones de seguridad, algo que suele ocurrir en la industria naviera. Por lo tanto, las características relacionadas con AIS y el clima son las únicas que se pueden utilizar para predecir el rendimiento del buque. Aunque solo se utilizan información meteorológica y dinámica, la SOG aún se puede predecir con precisión porque las RPM y el par motor generalmente no son volátiles durante el funcionamiento de los buques.

Aunque se ha encontrado que la SOG del buque se ve influenciada por otros factores mayormente que aquellos que representan las corrientes oceánicas, se ha decidido incluirlos dado que se sabe que la corriente oceánica afecta el rendimiento de un buque, y así evitar perder el efecto de la corriente oceánica mediantes sus características representativas.

6.3.4. Implementación, Resultados y Discusión de los Modelos Predictivos

Esta Sección describe las técnicas de modelado y el método general seguido para construir modelos potenciales de aprendizaje automático para la predicción de la velocidad del buque, como DTR y LR, y modelos de conjunto, como XGBR, RFR y ETR.

Para llevar a cabo la implementación de estos modelos se ha usado la librería scikit-learn 8.1.4 para cada uno de los modelos, exceptuando el XGBR en el que se usó la libería XGBoost 8.1.5. Para la elección de los hiperparámetros con los que construir los modelos propuestos se ha optado por los parámetros

preestablecidos. Sería interesante observar el trabajo que se propone para este apartado en M. Abebe et al., 2020 [10], en el que realizan un estudio de ajuste de los hiperparámetros del modelo mediante la optimización Bayesiana.

Para verificar la validez de los posibles modelos, se llevó a cabo un análisis de regresión con los datos de funcionamiento del buque a escala real y se obtuvieron los resultados de la regresión con el conjunto de datos de entrenamiento y de prueba. El enfoque clásico divide el conjunto de datos en dos conjuntos aleatorios, en adelante, el conjunto de entrenamiento y el conjunto de prueba. La proporción de división de los dos conjuntos de datos oscila entre 80/20 y 50/50, dependiendo del tamaño del conjunto [10]. En este caso, el conjunto de datos se dividió en un 60 % para el entrenamiento y un 40 % para las pruebas.

Para evaluar el rendimiento del modelo de manera más robusta y reducir el riesgo de sobreajuste, se aplica una técnica de validación cruzada k-fold. El conjunto de datos de entrenamiento se dividió en k submuestras, lo que significa que el modelo se ejecutará k veces de forma iterativa, utilizando k-1 submuestras para entrenar el modelo y el resto de las submuestras para las pruebas.

Después de cada iteración, se calcula la puntuación del modelo en el conjunto de pruebas. Al finalizar las 10 iteraciones, se obtiene una matriz de puntuaciones que representa el rendimiento del modelo en cada carpeta de la validación cruzada. Luego, se calcula el promedio de estas puntuaciones para obtener una medida consolidada del rendimiento del modelo, representada como la "mean cross validation score".

La utilización de la validación cruzada k-fold permite una evaluación más robusta del modelo y ayuda a detectar posibles problemas de sobreajuste. Al dividir el conjunto de datos de entrenamiento en múltiples submuestras, se proporciona al modelo la oportunidad de aprender de diferentes patrones y variaciones presentes en los datos, lo que puede mejorar su capacidad de generalización a nuevos datos.

Posteriormente el modelo es entrenado con todo el conjunto de datos de entrenamiento, y finalmente evaluado sobre el conjunto de prueba, con el cuál calculamos las métricas estadísticas de rendimiento.

Resultados

Para obtener una comparativa de los cinco modelos construidos con los datos de entrenamiento comentados en la Sección anterior, se atenderán a los criterios estadísticos explicados en el apartado 3.5.3, tiempo de ajuste y predicción, la importancia de las distintas variables independientes a la hora del ajuste y la relación de los datos de prueba y predicción. Esto queda recogido en la Tabla **6.5** en la que se prueban la precisión de los diferentes modelos en la validación cruzada de 10 "carpetas"/iteraciones; como sus rendimientos para los datos de test finales una vez entrenados con todos los datos en **6.6**

	DTR	**LR**	**XGBR**	**RFR**	**ETR**
Media [%]	98.206	19.963	96.107	98.362	98.476
Std. [%]	0.196	0.791	0.135	0.128	0.132
Min [%]	97.938	18.564	95.957	98.213	98.265
Mediana [%]	98.234	20.088	96.055	98.363	98.500
Max [%]	98.571	21.462	96.349	98.572	98.631
Tiempo de computación [sec]	10.75	2.68	200.64	794.44	265.64

Tabla 6.5: Estadística descriptiva de la precisión de los modelos en la validación cruzada de las 10 "carpetas".

Como se puede observar, de los cinco modelos planteados, el LR queda totalmente descartado para poder explicar el SOG con una precisión mucho menor que los otros cuatro modelos. De estos cuatro

modelos, DTR fue el que menos tiempo de cálculo necesitó, con una precisión bastante elevada, que los modelos de conjunto basados en DTR mejoran algo más dicha precisión. En cuanto a las técnicas de ensemble, el bagging proporcionó un mejor resultado que el boosting.

Otra forma de evaluar los modelos es a través de las precisiones obtenidas para el conjunto de datos de prueba, una vez entrenados los modelos. En la Tabla **6.6** se observa como los modelos basados en conjunto de DTR continúan demostrando el mayor coeficiente de determinación (R^2) y una de las variabilidades más bajas. A pesar de la alta precisión alcanzada por estos modelos, dado que el modelo de DTR no se distancia significativamente en términos de variabilidad respecto a los mejores modelos y presenta resultados de precisión y computacionales sobresalientes, se ha elegido como la opción preferida.

Modelo	R^2	**MAE**	**MSE**	**RMSE**	**Tiempo de computación [sec]**
LR	0.2046	2.3824	9.3817	3.0630	0.205
DTR	0.9823	0.1801	0.2081	0.4562	1.115
XGBR	0.9606	0.4288	0.4640	0.6812	19.9093
RFR	0.9839	0.1816	0.1894	0.4352	87.2759
ETR	0.9851	0.1758	0.1753	0.4187	30.8716

Tabla 6.6: Rendimiento del modelo en el conjunto de datos de prueba.

Por último, se presentan los resultados de los diversos modelos en términos de la importancia atribuida a cada una de las variables en el modelo, así como la concordancia entre las predicciones y los valores reales en los datos de prueba. En estos resultados, se observa que el modelo seleccionado, DTR, comparte la relevancia de las variables con los mejores modelos. Las variables explicativas más relevantes incluyen parámetros del buque,así como las condiciones meteorológicas; siendo Draft, resist,bottomT, so, zos y VTM10 algunas de las más relevantes. Estas variables desempeñan un papel significativo en la capacidad del modelo para hacer predicciones precisas.

LR

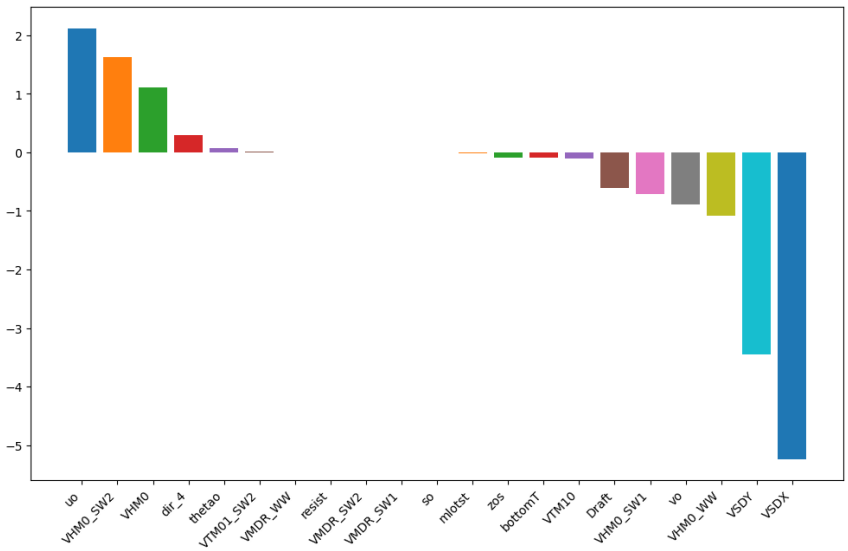

Figura 6.5: Importancia de las diferentes variables independientes para el modelo LR.

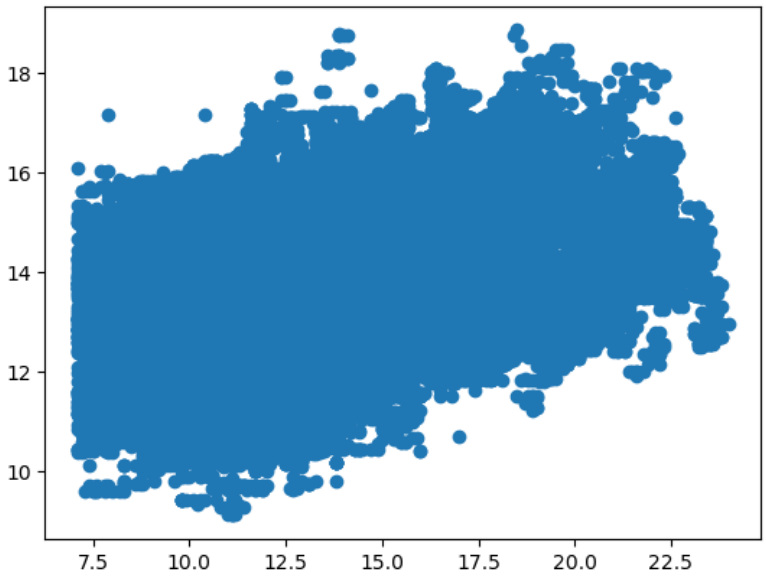

Figura 6.6: Relación entre la predicción y el valor actual del SOG en los datos de test para el modelo LR.

DTR

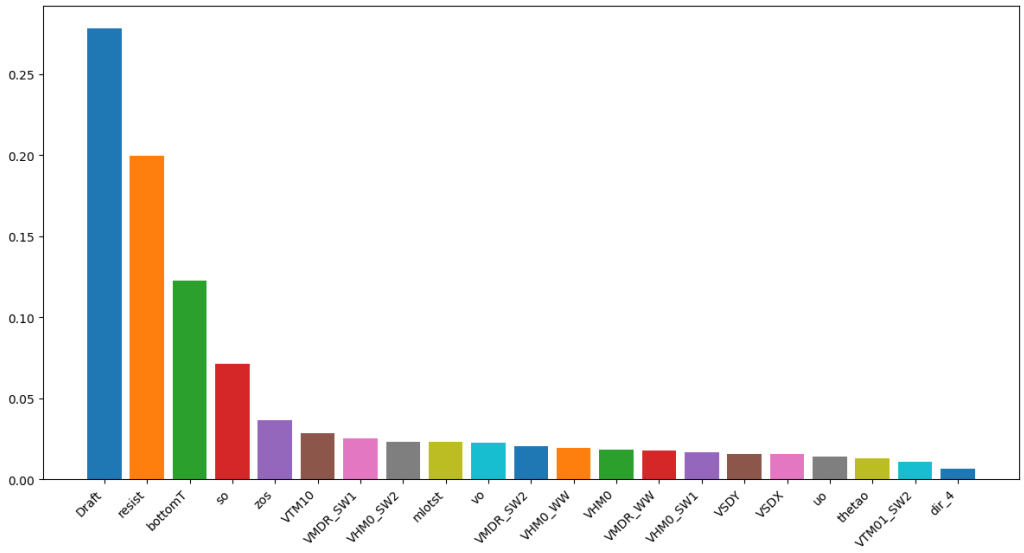

Figura 6.7: Importancia de las diferentes variables independientes para el modelo DTR.

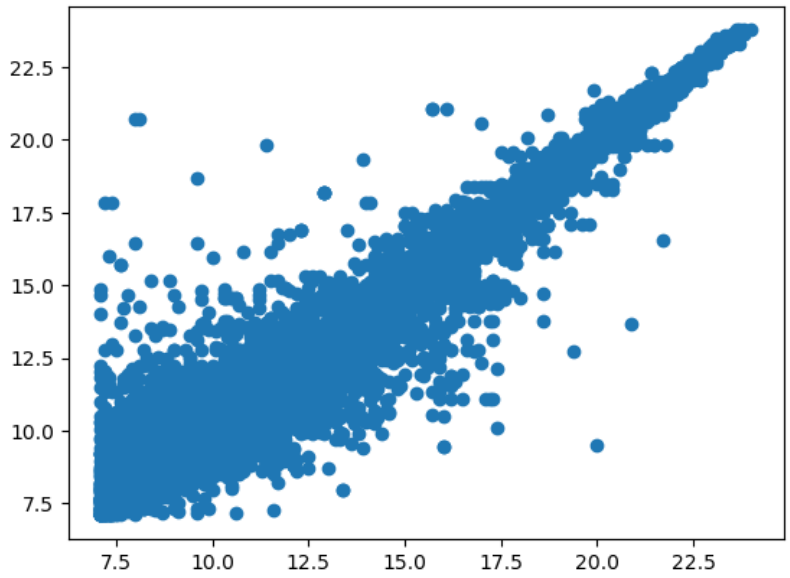

Figura 6.8: Relación entre la predicción y el valor actual del SOG en los datos de test para el modelo DTR.

XGBR

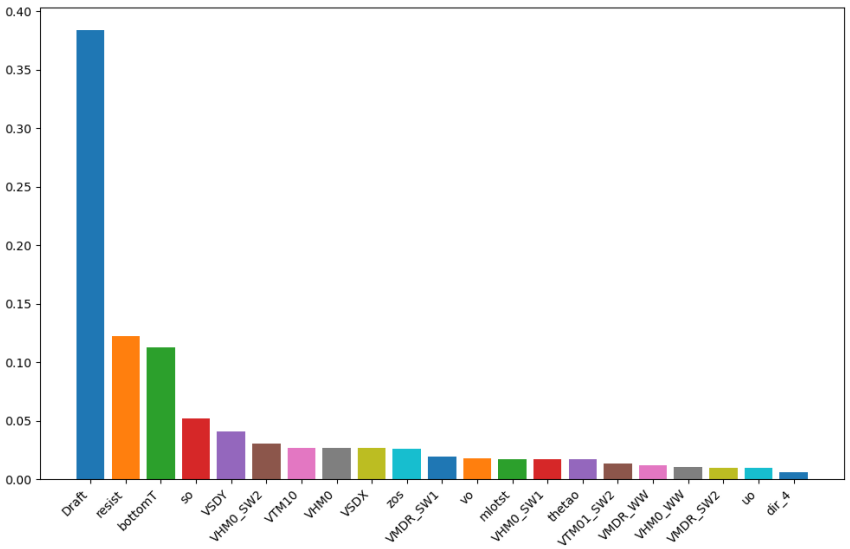

Figura 6.9: Importancia de las diferentes variables independientes para el modelo XGBR.

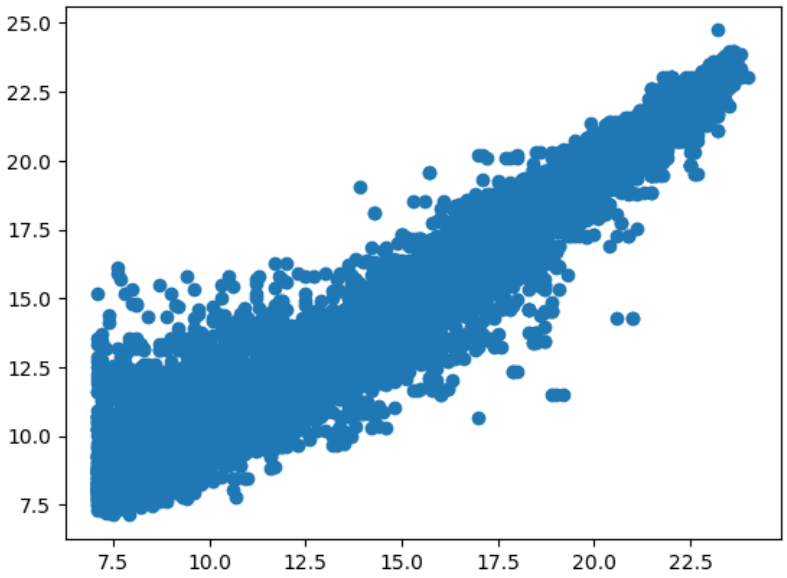

Figura 6.10: Relación entre la predicción y el valor actual del SOG en los datos de test para el modelo XGBR.

RFR

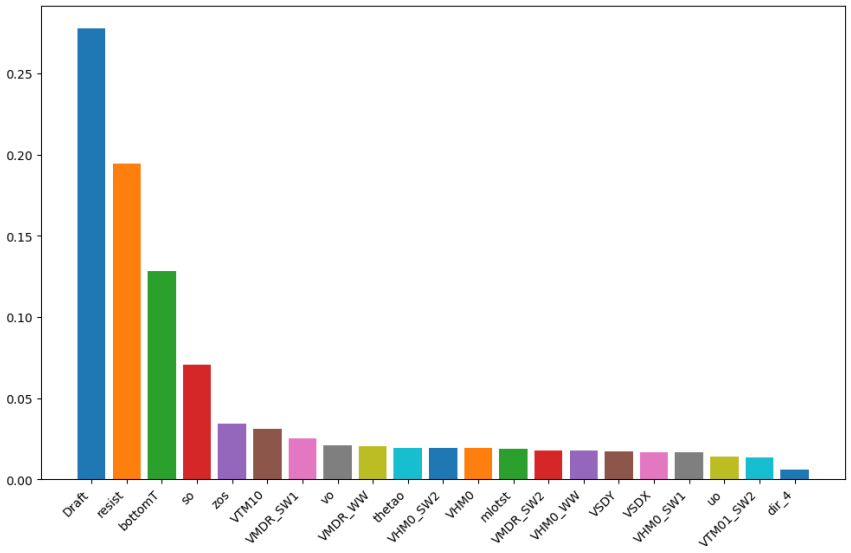

Figura 6.11: Importancia de las diferentes variables independientes para el modelo RFR.

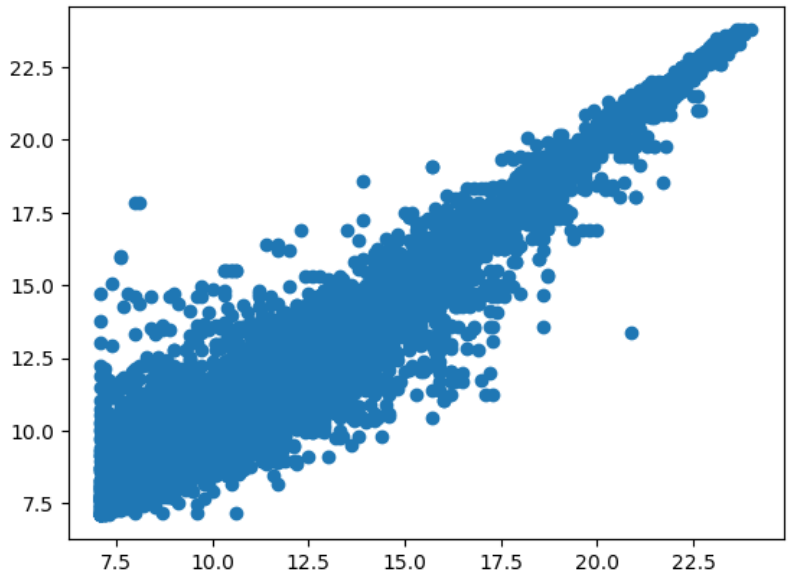

Figura 6.12: Relación entre la predicción y el valor actual del SOG en los datos de test para el modelo RFR.

ETR

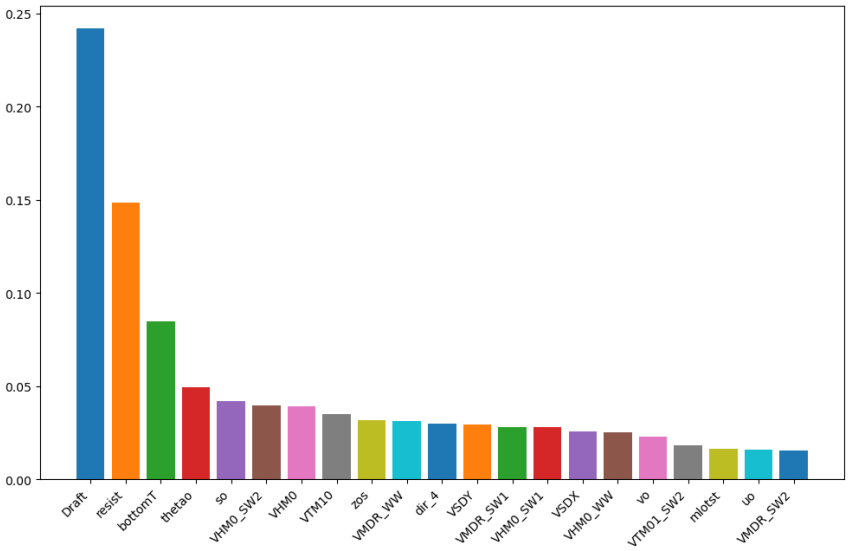

Figura 6.13: Importancia de las diferentes variables independientes para el modelo ETR.

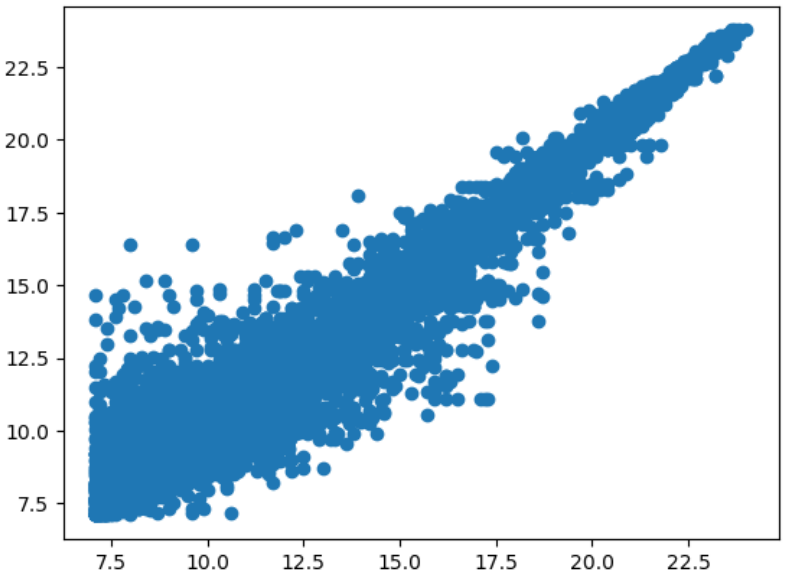

Figura 6.14: Relación entre la predicción y el valor actual del SOG en los datos de test para el modelo ETR.

Capítulo 7

Construcción del modelo

7.1. Introducción

Como se ha comentado en la introducción, la reducción del consumo de combustible es muy importante en la navegación de los buques. Para minimizar el consumo de combustible, sería útil ajustar la potencia del motor y viajar con condiciones meteorológicas y de oleaje eficientes a lo largo de la ruta. Sin embargo, se trata de un planteamiento ingenuo, ya que los buques tienen que ceñirse a las franjas horarias y los puertos de destino reservados. Además del consumo de combustible hay que asegurarse de que la ruta se elige en función del límite de tiempo y el destino. Esto da lugar a dos objetivos que queremos optimizar con el enrutamiento:

- minimizar el consumo de combustible
- minimizar la diferencia entre la hora de llegada y la hora reservada en puerto

Para poder calcular rutas en función de factores como el consumo de combustible y la meteorología, es necesario disponer de los datos correspondientes. En este Capítulo ilustramos cómo se utilizaron posteriormente los datos para el cálculo de la ruta en forma de un modelo espacial.

Según nuestro primer objetivo, minimizar el consumo de combustible, la consecuencia obvia sería modelizar el consumo de combustible como variable objetivo. Sin embargo, como no se disponía de suficientes datos para modelizar el consumo de combustible o incluso la potencia del motor, se partió del modelo empírico de *WinGD (2016)* [13] y decidimos obtener este valor a partir del tiempo, condicionado por la velocidad del buque, y la constante de fuel, tal cómo se planteó en la Eq. 3.11.

Para hacer frente al segundo de los objetivos, y junto a lo planteado en el Capítulo 6 podemos calcular el tiempo que lleva recorrer la ruta. Para ello debemos modelizar el espacio marítimo para poder saber en cada punto el tiempo que conlleva pasar por allí basado en la distancia a recorrer y la velocidad sobre el suelo (SOG), esta última, condicionada por las condiciones marítimas.

En las siguientes secciones se exponen en detalle el framework de planificación de rutas marítimas multicriterio desarrollado para abordar los criterios del problema. Como se muestra en la Fig. 7.1, el framework consta de seis partes: criterios de optimización, análisis predictivo de la velocidad del buque, construcción del modelo, algoritmo multicriterio, evaluación de rutas y selección de rutas.

Los criterios de optimización incluyen el tiempo de navegación y el consumo de combustible. El tiempo de navegación puede obtenerse sumando el tiempo asociado a cada segmento de la ruta. El consumo de

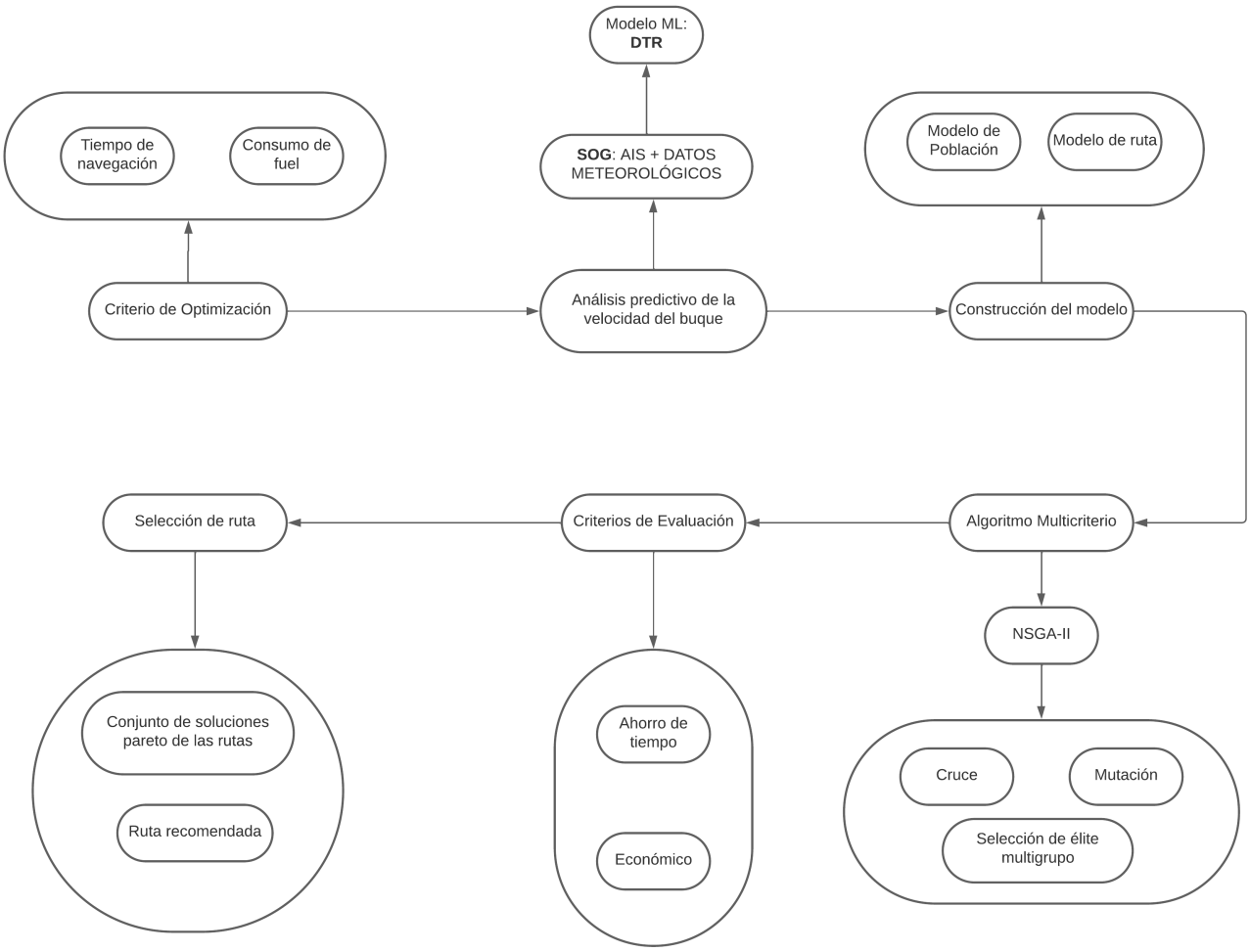

Figura 7.1: Framework de planificación de rutas multicriterio propuesto

combustible se simula mediante parámetros como el consumo por kWh y el tiempo de navegación, a su vez condicionado por la velocidad del buque. La predicción mediante un modelo de machine learning, *decision tree regressor* (DTR), se estudia en la parte de análisis de la velocidad del buque.

La parte de construcción del modelo incluye la construcción del modelo de ruta y el método de codificación de la población. El algoritmo multicriterio incluye la operación de cruce, operación de mutación y selección de élite multigrupo. Los principales criterios de evaluación de la ruta son el ahorro de tiempo y el económico.

La selección de rutas consta de dos partes: una parte consiste en proporcionar múltiples conjuntos de soluciones de rutas obtenidas mediante la optimización de algoritmos, mientras que la otra parte tiene por objetivo proporcionar la ruta recomendada que mejor satisfaga los requisitos basados en valores personalizados.

7.2. Cuadrícula de SOG: AIS + datos meteorológicos

El trabajo realizado en la Sección 6.3 nos sirve para modelizar las características del mar en un tiempo específico según las características del barco y la meteorología. Para la comprensión de estos datos en una ruta completa y obtener nuestro objetivo de calcular el coste asociado a esta, se modela el espacio

del dominio marítimo en un mapa de retículas (malla de rejillas o, en inglés, grid mesh) rectangular con espacio constante en las coordenadas naturales, 1/12° de resolución tanto en longitud como latitud (aproximadamente unos 8km), siguiendo la definición del grid estándar proporcionado por los datos de Copernicus [11]. Esto da lugar a la siguiente definición de grid:

GLOBAL (180°W-180°E; 89°S–90°N): 1/12º, regular grid, 4320 x 2041 [11]

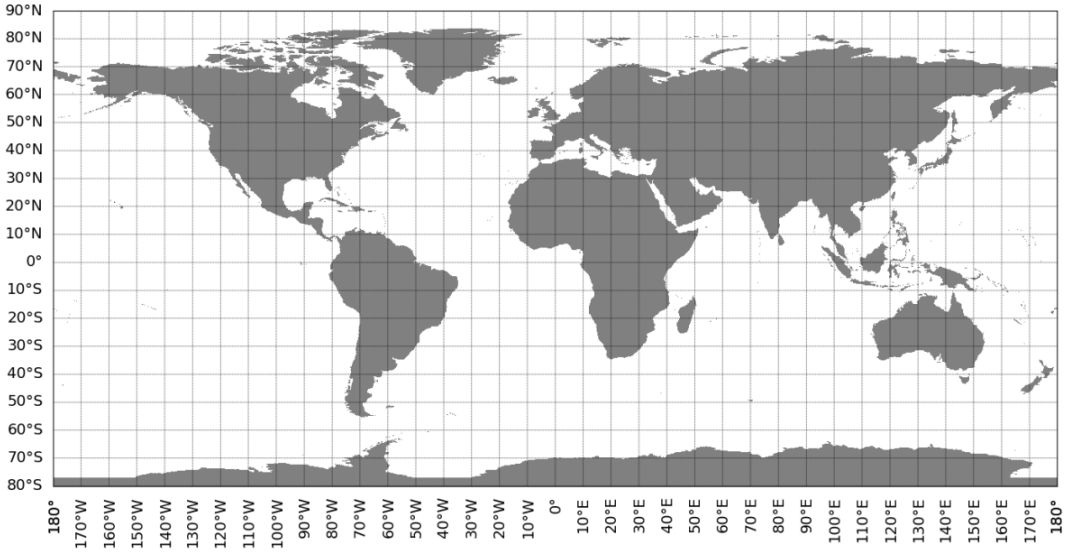

Figura 7.2: Mapa de retículas de 10° Fuente: [11]. Nuestra modelización será de 1/12°(0.083°)

Tomando esta referencia nuestro siguiente paso ha sido asignar a cada cuadrícula la velocidad asociada sobre el suelo (SOG). Para ello descargamos las condiciones meteorológicas marítimas de Copernicus, mediante el uso de *PyDAP* 8.1.2, del cuál obtenemos todo el archivo de datos desde 2021. Con ello y junto al modelo predictivo DTR discutido en el Capítulo 6, somos capaces de asignar a cada celda el valor predicho para ese barco y las condiciones meteorológicas que se encuentra en un tiempo dado.

Como podemos deducir este grid muestra una foto fija en un instante de tiempo. Por lo que deberá actualizarse a medida que avance la ruta, y por tanto el incremento del tiempo a medida que avanza. Este tiempo de actualización depende de los tiempos de actualización de los datasets meteorológicos de Copernicus. Como se ha estudiado en el Capítulo 6 el tiempo de actualización es de 3 horas para *Global Ocean Waves Analysis and Forecast* y 1 día para *Global Ocean Physics Analysis and Forecast*; por lo que se tomaría este último tiempo como referencia para actualizar los valores de cada celda de la cuadrícula a medida que avanza en ese tiempo la ruta.

Finalmente, esta idea de actualización a medida que avanza la ruta, que sería la ideal, no se llevó a cabo debido a la complejidad computacional y temporal de dicho proceso por cada ruta que forma parte de la población. Por lo que se decidió tomar el instante de tiempo inicial de salida de la ruta como instante de calculo de los valores de la cuadrícula para toda la ruta. Esto permitiría en un ambiente real poder volver a ejecutar el algoritmo pasado un día o pasado unas ciertas horas con las nuevas coordenadas de salidas actualizadas a ese momento y volver a obtener un nuevo grid que nos proporcionaría las siguientes mejores coordenadas de ruta para optimizar nuestro criterio.

Teniendo esto en cuenta, tomamos los valores climatológicos en cada celda del grid y mediante el modelo DTR construido y los datos del barco predecimos la velocidad SOG que se llevaría en ese punto. También consideramos la dirección de navegación, aunque su influencia en el modelo DTR es relativamente baja, como se observó en las importancia de las variables a la hora de construir el modelo. Sin embargo, es importante destacar que esta variable de dirección adquirirá un papel más significativo en etapas posteriores del cálculo de tiempo.

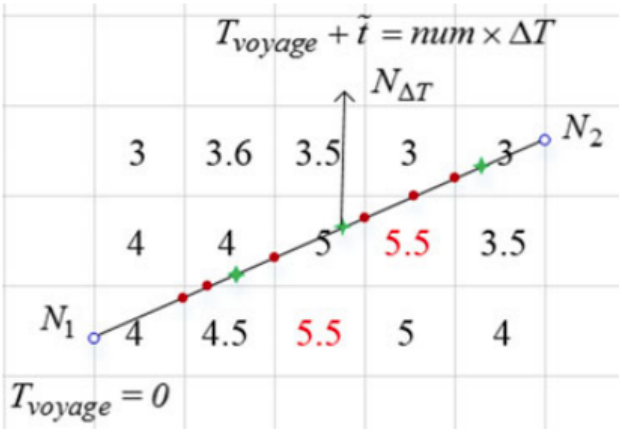

Figura 7.3: Ruta dividida según el tamaño de la retícula de los datos meteorológicos. El punto verde indica la actualización de los datos para esa ruta siendo $T_{viaje} + \tilde{t}$, siendo múltiplo del periodo de tiempo de actualización de 1 día como ΔT. Fuente: [3].

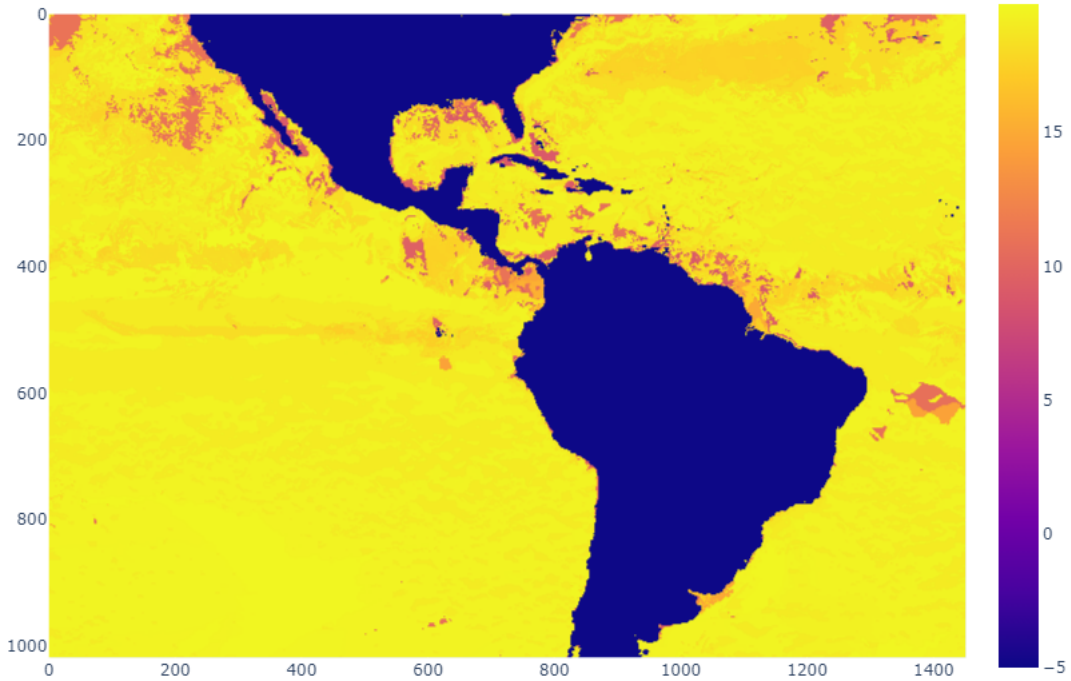

Figura 7.4: Modelado del SOG para la dirección Oeste en la zona del Caribe para el día 25/06/2021 a las 12:00

Como se comentaba en el párrafo anterior, la importancia a la hora de modelizar el SOG con nuestro modelo no es tan apreciable. Pero para nuestro objetivo de minimizar el tiempo la diferencia entre la hora de llegada y la hora reservada en el puerto, es decir minimizar el tiempo de navegación para ajustarse a la llegada, la dirección juega un papel clave.

La determinación de la distancia entre dos ubicaciones se estima mediante la aplicación del teorema de Pitágoras a las distancias en dirección norte-sur y este-oeste.

Las líneas de latitud (los diferentes paralelos), alineaciones circulares concéntricas, experimentan una disminución progresiva en su tamaño a medida que se acercan a los polos. Estas líneas convergen en un único punto en los polos, donde se originan los meridianos. En la línea ecuatorial, se estima que un grado

de longitud corresponde aproximadamente a 111.321 kilómetros, mientras, a una latitud de 60 grados, un grado de longitud se reduce considerablemente a unos 55.802 kilómetros (este cálculo se basa en el modelo de la esferoide Clarke 1866) [12]. Un grado de paralelo equivale a $111 * \cos w$, es decir desde los 111 kilómetros en el Ecuador, hasta los 0 en los polos. Un grado meridiano equivale siempre a 111 kilómetros [112].

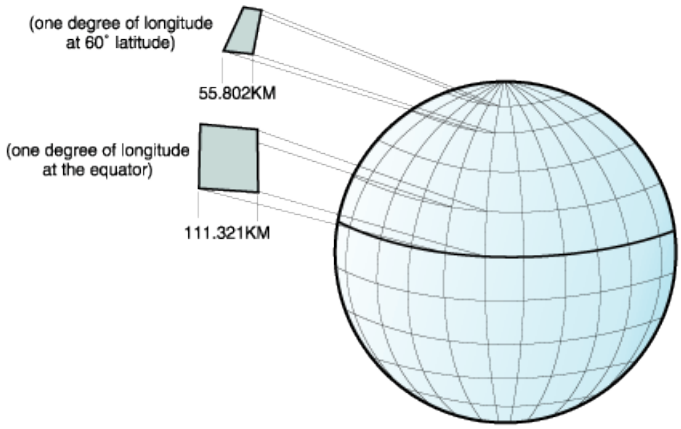

Figura 7.5: Diferencias de dimensión de km de longitud entre diferentes ubicación geográficas. Fuente: [12]

Esta traslación del coste de recorrer un kilómetro según la dirección del barco, ya sea para tomar una línea de latitud o un grado meridiano quedan modeladas en el script desarrollado *notebooks/gridLength.ipynb*[1] y podemos ver su resultado para el grado de paralelo en la imagen 7.6:

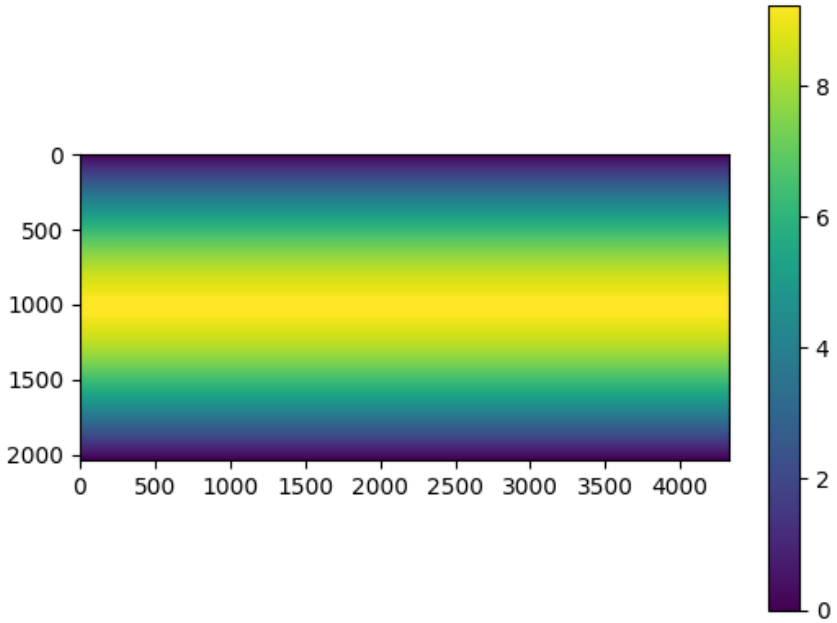

Figura 7.6: Modelización coste grado de latitud en el grid planteado (dirección EW).

Finalmente esto nos lleva a poder construir el grid en base a nuestros criterios de minimizar el tiempo, para saber cuánto tiempo necesita el barco para navegar a través de esa celda en minutos. Por lo tanto, dividimos la longitud real de cada celda en metros por la velocidad en metros por minuto. Como el SOG predicho por nuestro modelo viene dado en nudos (knots), lo convertimos en metros por minutos multiplicándolo por 30.87.

[1] `https://gitlab.inf.uva.es/migchav/tfg-miguel-chaveinte/-/blob/master/notebooks/gridLength.ipynb`

Este proceso de modelado del grid se lleva a cabo en las funciones del siguiente fichero *predict_sog.py* [2]. Los resultados de la conjunción de la predicción del SOG y la diferencia notable que produce la dirección en ello a la hora de establecer valores para el grid es apreciable, como se demuestra en la imagen 7.7:

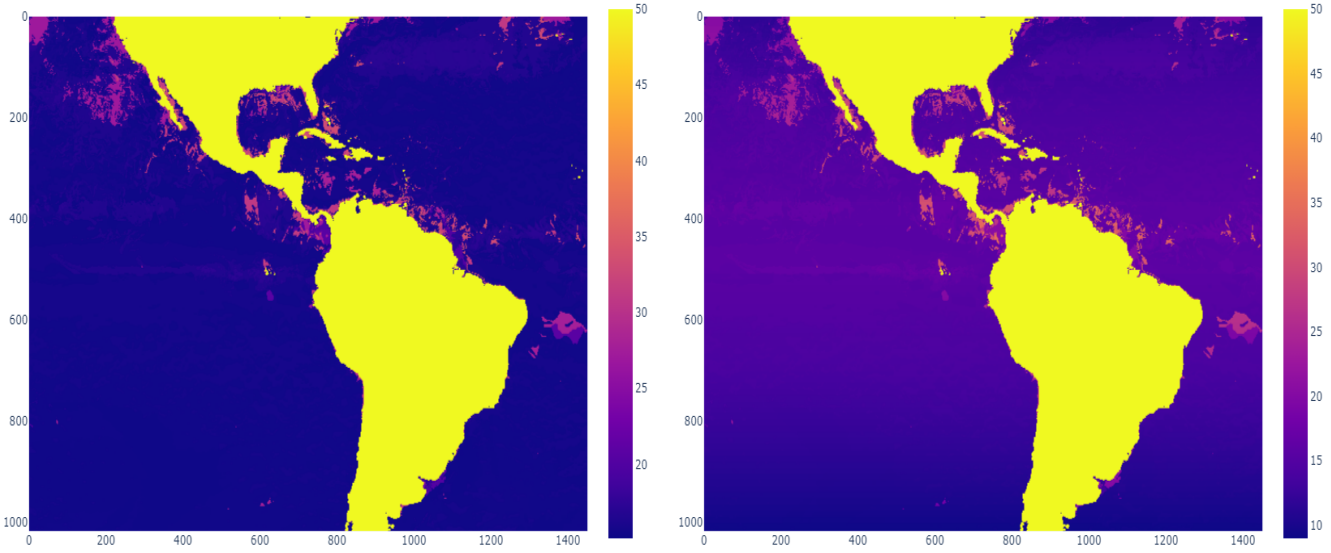

Figura 7.7: Modelado del grid con la conjunción del SOG y los grados de latitud (distancia y dirección). A la izquierda se muestra el modela para la dirección sur y a la derecha para la dirección este; resaltando sobre los puntos [200,600] con diferencias significativas de coste en tiempo de trayecto para cada dirección norte-sur / este-oeste.

7.3. Funciones Objetivo

La población es por tanto evaluada según las funciones objetivo definidas, que en nuestro caso estos criterios se basan en la definición de los objetivos a minimizar: la diferencia de tiempo y el gasto de fuel.

Para la implementación de las funciones de aptitud (fitness), el primer paso era calcular el grid, anteriormente definido en la Sección 7.2, el cual contiene el tiempo necesario para atravesar una celda. Con ello obtenemos el tiempo por cada celda por la que pasa y calculamos la diferencias entre el tiempo de llegada esperado y el tiempo calculado por la ruta, teniendo en cuenta la dirección de la ruta a la hora de atravesar la celda. Se puede observar el cálculo del tiempo, tomando el grid, en la función *time_differences* [3].

El consumo de combustible de los buques es un factor clave para determinar su eficiencia energética y su impacto ambiental. No obstante, debido a la complejidad inherente a este proceso, no existe una metodología única y precisa que permita estimar dicho consumo, ya que depende de múltiples variables, como el tipo de buque, la ruta, la velocidad, el estado del mar, la carga, el tipo de combustible, etc. Tras realizar una investigación para calcular el modelado del coste, se descubrió que la mayoría de los buques no utilizan el 100 % de la potencia de sus motores; la mayoría utilizan entre el 70 % y el 85 % [113]. Tomando como **70 % la potencia del motor** a lo largo de toda la ruta, y el modelo empírico propuesto en *WinGD (2016)* [13] (visible en la figura 7.8), el consumo de combustible para una potencia de motor de 0.7 es de 5,11 toneladas. Para nuestros cálculos esto resulta en $154 g/kWh$. A continuación,

[2] `https://gitlab.inf.uva.es/migchav/tfg-miguel-chaveinte/-/blob/master/src/modelo/predict_sog.py`

[3] `https://gitlab.inf.uva.es/migchav/tfg-miguel-chaveinte/-/blob/master/src/utilsAlgoritmo/func_objetivos.py`

se calcularon los valores para un motor de ejemplo, que necesita $33200kW$. Se utilizan para ello las Eq. 3.11 y 3.12 definidas en el Capítulo 3.

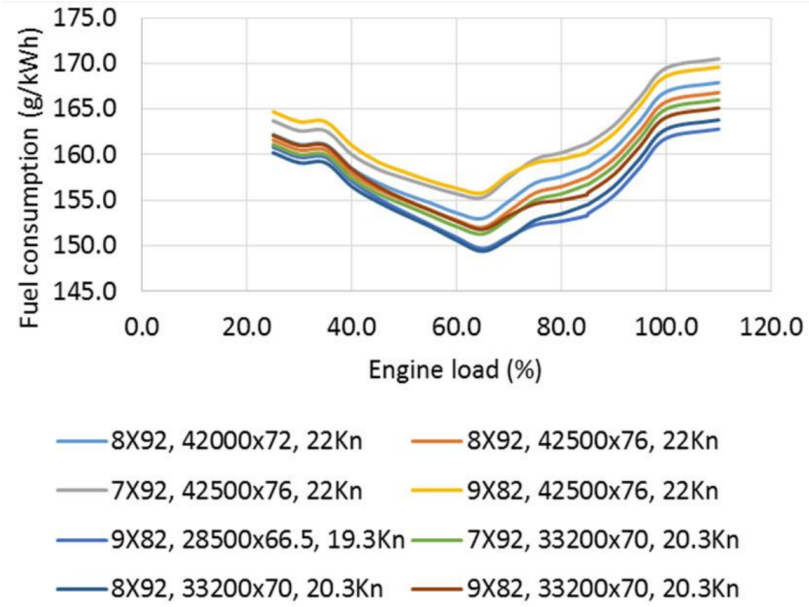

Figura 7.8: Modelo de consumo de fuel junto a la carga del motor usada. Fuente: [13]

7.4. Modelo de población e inicialización de la zona de navegación

Para el establecimiento de la zona de navegación se toma la ruta ortodrómica (great circle route) como ruta de referencia. La ruta ortodrómica es una ruta ideal debido a su corto trayecto, lo que reduce el tiempo de navegación [114].

La población tomada para los AGs comprende múltiples individuos, y cada individuo está representado por una serie de coordenadas de latitud y longitud. En la Eq. 7.1 se representa una ruta marítima, donde X_j es un vector bidimensional que contiene valores de longitud y latitud.

$$X = [X_0, X_1, \ldots, X_j, \ldots, X_{M-1}, X_M] \tag{7.1}$$

Cada ruta puede generarse en un área de búsqueda limitada basada en la ruta de referencia. El área de búsqueda limitada zona de búsqueda limitada es la zona ampliada a ambos lados de la ruta de referencia basada en las características históricas de la ruta del buque. Como se muestra en la Figura 7.9, S y E son los puntos inicial y final, respectivamente, de la ruta de referencia. A partir de la ruta ortodrómica como ruta de referencia entre estos dos puntos, y el área encerrada por la línea con guiones y punteada es el área de búsqueda del punto de paso del barco. El límite superior del área de búsqueda es el $UpperBound$, mientras que el límite inferior es el $LowerBound$, que se representan mediante las ecuaciones:

$$UpperBound = \left[\begin{array}{ccccccc} \lambda_s & UpperBound_1 & \ldots & UpperBound_i & \ldots & \lambda_d \end{array} \right], \quad i = \overline{1,n} \tag{7.2}$$

$$LowerBound = \left[\begin{array}{ccccccc} \lambda_s & LowerBound_1 & \ldots & LowerBound_i & \ldots & \lambda_d \end{array} \right], \quad i = \overline{1,n} \tag{7.3}$$

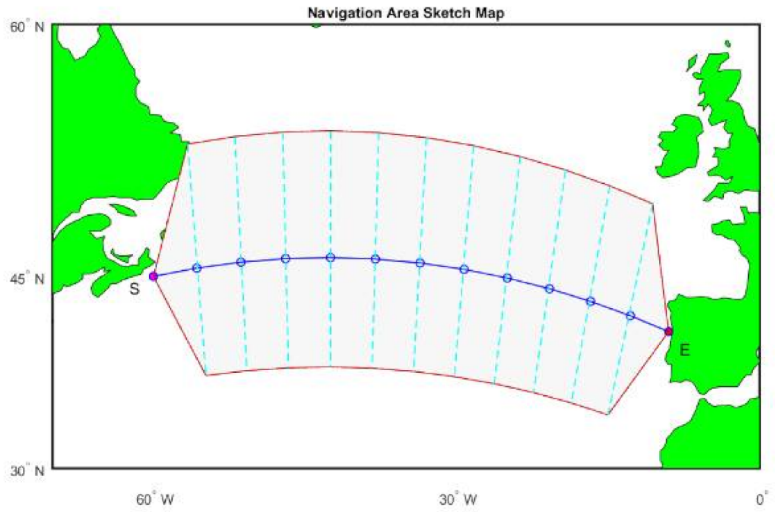

Figura 7.9: Zona de navegación población. La ruta ortodrómica es la ruta de referencia. Fuente: [14]

Por lo que la ruta queda representada en forma de cromosoma. Supongamos que el tamaño de la población es M. Sea el vector $lonpts$ que representa la información de longitud del cromosoma y el vector $latpts_j$ que representa la información de latitud del j-ésimo cromosoma. La longitud para cada cromosoma es la misma, mientras que la latitud para cada cromosoma debe estar dentro del rango de $[LowerBound, UpperBound]$.

$$\begin{aligned} lonpts &= \left[\begin{array}{ccccccc} l_s & l_1 & \dots & l_i & \dots & l_d \end{array}\right], \quad i = \overline{1,n} \\ latpts_j &= \left[\begin{array}{ccccccc} \lambda_s & \lambda_{j1} & \dots & \lambda_{ji} & \dots & \lambda_d \end{array}\right], \quad i = \overline{1,n}; j = \overline{1,M} \end{aligned} \tag{7.4}$$

El número de cromosomas necesarios se genera aleatoriamente en función del tamaño de la población. Por ejemplo, un cromosoma individual puede generarse mediante:

$$latpts_1 = rand(1,N) \times [UpperBound - LowerBound] + LowerBound \tag{7.5}$$

donde N es la dimensión de las variables (el número de waypoints), rand(1,N) genera un vector de números aleatorios con una longitud de N, que se distribuye uniformemente entre 0 y 1.

En la Fig. 7.10 se representan un conjunto de 10 rutas población en nuestro sistema, marcadas por la zona de navegación discutida y la ruta *great circle*.

A continuación se presentan las ideas de cruce y mutación, que fueron tomadas de *Niu et al. (2016)* [115], adaptándose a las condiciones de nuestro problema:

7.5. Cruzamiento / Recombinación

Para llevar a cabo el cruce, decidimos cuántas nuevas rutas deben construirse. Para ello, seleccionamos pares de dos rutas y combinamos estas para dar lugar a dos nuevas rutas.

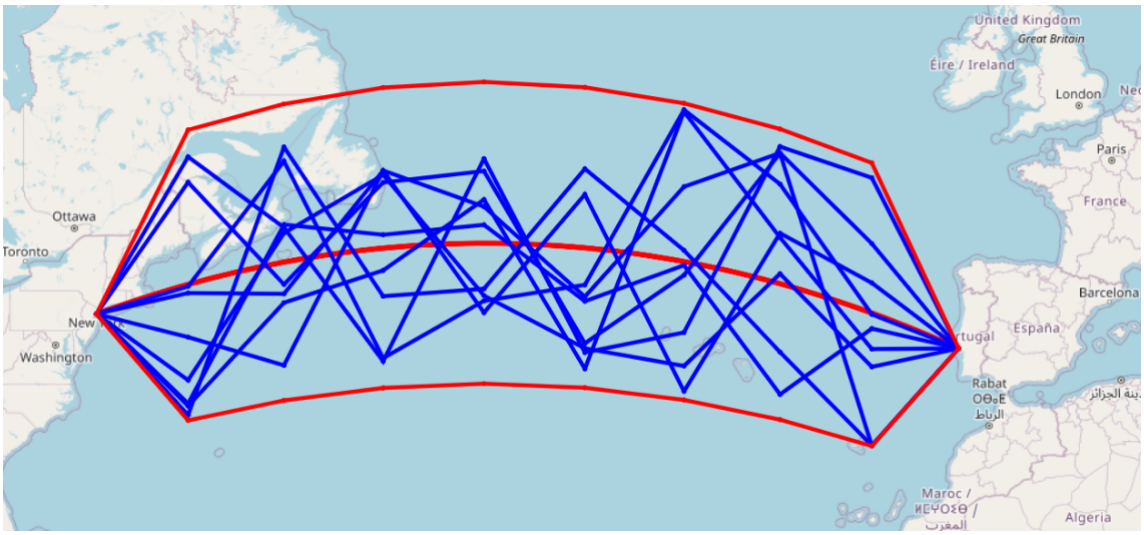

Figura 7.10: Población de tamaño 10 rutas con 10 waypoints cada una de las rutas. La ruta roja central representa el great circle, y las rutas rojas superior e inferior los límites de la zona de navegación.

El punto de cruce se selecciona aleatoriamente, siguiendo la operación tradicional de cruce de un solo punto y, a continuación, se generan dos individuos descendientes intercambiando las partes derechas de los dos cromosomas parentales. Siendo dos cromosomas de longitud fija, se seleccionan dos nodos de cruces, no siendo estos los nodos de inicio ni final en los dos cromosomas, y se genera un camino auxiliar tomando estos como nuevos puntos iniciales y finales. Luego, los cromosomas descendientes se generan agregando el camino auxiliar a los cromosomas parentales cuyas partes derechas se intercambian entre sí.

En nuestro caso, el método de generación de este camino auxiliar del cruce se ejecuta de manera similar al de la población inicial. Se toman esos nodos como puntos de inicio y final, y basándonos en la ruta ortodrómica entre esos dos puntos y en los *UpperBound* y *LowerBound* reducidos (se reducen a la mitad para que este camino auxiliar quede compacto y no genere coordenadas muy distantes entre ellas) se genera una nueva alrededor de ella.

En la figura 7.11, se puede apreciar el concepto comentado anteriormente. La ruta azul oscura y naranja oscura son los dos cromosomas parentales sometidos al cruce. En estos se calcula el nodo de cruce inicial (punto rojo) y un punto final (verde) en cada ruta y calculamos el camino auxiliar; dando esto la ruta naranja y azul claras.

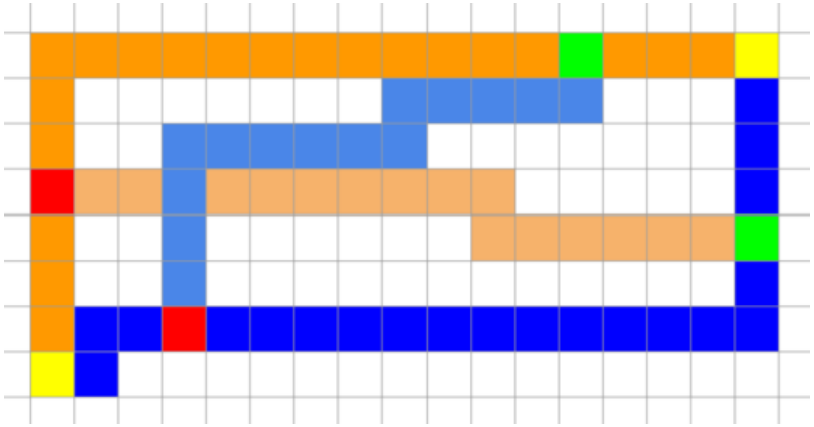

Figura 7.11: Concepto Cruzamiento rutas.

7.6. Mutación

Para el concepto de mutación se lleva a cabo un proceso similar al anterior presentado en el *Cruzamiento*: se calculan dos puntos de mutación de forma aleatoria, como nodo inicial y final; y de la misma forma generamos el camino auxiliar nuevo de mutación entre ambos puntos. En la figura 7.12 se presenta de igual manera el proceso. La ruta azul oscura es la ruta a mutar; y de nuevo, el punto rojo es el nodo inicial, y el verde el final para el camino auxiliar de mutación,dando lugar al nuevo camino auxiliar y nueva ruta representado por la ruta azul clara.

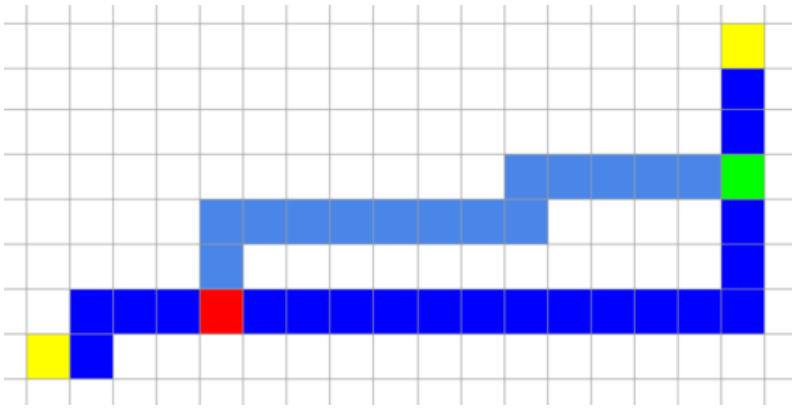

Figura 7.12: Concepto mutación rutas.

7.7. Implementación: Pymoo y parámetros de entrada

La implementación de los pasos del algoritmo NSGA-II descritos se han llevado a cabo bajo el uso de la librería *Pymoo* 8.1.6. Este framework esta diseñado para ser extendido y modificado según nuestro problema gracias a su diseño modular.

7.7.1. Implementación del problema

Para ello uno de los primeros pasos sería instanciar el elemento problema para definir su implementación. Definimos un nuevo objetivo, que viene heredado de la clase *Problem*, con el establecimiento de los atributos de nuestro problema, como el número de variables a tener en cuenta (*n_var*), número de objetivos (*n_obj*) y restricciones (*n_constr*) y los límites inferior (*xl*) y superior (*xu*). La función encargada de la evaluación es *_evaluate* cuya implementación viene dada por la funciones objetivo definidas con anterioridad.La interfaz de la función son los parámetros *X* y *out*. En esta implementación por elementos, *X* es una matriz unidimensional NumPy de longitud *n_var* que representa una única solución que se va a evaluar. La salida debe escribirse en el diccionario *out*. Los valores objetivo deben escribirse en out["F"] como una lista de matrices NumPy con una longitud de *n_obj* [116].

7.7.2. Inicialización del algoritmo

Una vez definido el problema, siguiendo con el enfoque orientado a objetos de *Pymoo*, tenemos que definir un objeto algoritmo a continuación. El algoritmo seleccionado, que se encuentra disponible en la librería, es el algoritmo multi-objetivo *NSGA-II*(../algorithms/moo/nsga2.ipynb). La librería permite elegir los hiperparámetos por defecto o crear la propia versión del algoritmo modificando sus componentes

internos. En la instanciación del objeto algoritmo se define el tamaño de la población y el número de offspring (número de descendientes). En este trabajo se ha realizado una adaptación del algoritmo para que sea compatible con el problema de las rutas marítimas, mediante la implementación de los siguientes módulos:

7.7.2.1. Población

Esta nueva clase hereda de la clase *Sampling*. Mediante el método constructor de la clase inicializamos los atributos de nuestro objeto Población. Para nuestro caso este debe contar con los atributos introducidos por el usuario de punto de partida, punto destino, UpperBound, LowerBound y waypoints en la ruta.

Con estos parámetros ya somos capaces de instanciar a nuestra función objetivo población, que realiza el trabajo según lo especificado anteriormente; devolviendo una lista de tamaño de la población introducido por el usuario, con cada una de las rutas representadas de la siguiente forma [[lon_coords],[lat_coords]].

7.7.2.2. Cruce

Esta nueva clase hereda de la clase *Crossover*. Mediante el método constructor de la clase inicializamos los atributos de nuestro objeto Cruce. Para ello contamos con los atributos UpperBound, LowerBound y waypoints en la ruta, introducidos por el usuario. Dentro del constructor tenemos que definir la decisión tomadas en la Sección de Cruzamiento 7.5, en las que tomábamos dos rutas como padres, para generar otras dos rutas y además con una probabilidad de cruce del 100 %.

Para la adaptación a nuestro modelo de ruta ([[lon_coords],[lat_coords]]) tuvimos que hacer un ajuste en la función *do* de la clase Crossover. Por lo que mediante un *override* a esta función ajustamos el *shape* (dimensión) de las rutas a devolver.

Una vez que seleccionamos los dos padres para hacer el proceso de cruce, llamamos a la función que implementa la metodología propuesta en la Sección Cruzamiento 7.5. Esto nos devuelve dos hijos, que serán los que formen parte de la nueva población.

7.7.2.3. Mutación

Esta nueva clase hereda de la clase *Mutation*. Mediante el método constructor de la clase inicializamos los atributos de nuestro objeto Mutación. Para ello contamos con los atributos UpperBound, LowerBound y probabilidad de mutación, introducidos por el usuario.

Con estos parámetros se lleva a cabo el proceso de mutación definido en la Sección 7.6, con cierta probabilidad.

7.7.3. Definición del criterio de terminación

Además, es necesario definir un criterio de terminación para iniciar el procedimiento de optimización. La forma más habitual de definir la terminación es limitando el número total de evaluaciones de la función o simplemente el número de iteraciones del algoritmo. En nuestro caso este viene marcado por el número de generaciones introducidas por el usuario.

7.7.4. Optimización

Por último, resolvemos el problema con el algoritmo y la terminación que hemos definido. La interfaz funcional utiliza el método *minimize*.

Cuando el algoritmo ha terminado, la función minimizar devuelve un objeto *Resultado*, que es lo que devuelve la función *runNSGA2Algorithm*[4]. Este objeto esta formado por los elementos X y F, comentados en la la Sección 7.7.1, siendo X las rutas del frente de Pareto óptimo y F es el vector de valores de las funciones objetivo evaluadas en X.

[4] `https://gitlab.inf.uva.es/migchav/tfg-miguel-chaveinte/-/blob/master/src/algoritmo/algorithmNSGA2.py`

Capítulo 8

Implementación

8.1. Tecnología utilizadas

En esta Sección se recogen las tecnologías y herramientas utilizadas que destacan en el desarrollo del proyecto.

8.1.1. MOTU

MOTU [117] es un cliente Python que permite el acceso a la descarga de los productos de datos oceanográficos y atmosféricos desde servidores remotos. MOTU significa Monitoring Oceanographic and aTmospheric daUta.

La API de MOTU en Python permite recuperar productos específicos mediante consultas y parámetros personalizados, así como filtrar los productos según parámetros como la fecha y la ubicación geográfica. MOTU mejora la eficiencia al aprovechar la potencia de procesamiento de la máquina y descargar múltiples productos de datos simultáneamente.

Se ha usado MOTU para llevar a cabo las descargas de los datos marítimos que permitieron, junto a los datos del AIS, construir un modelo DTR para modelizar la velocidad SOG.

8.1.2. PyDAP

PyDAP [118] es una biblioteca de Python que utiliza el protocolo OPeNDAP para acceder y manipular datos científicos. PyDAP permite a los desarrolladores interactuar con servidores que admiten OPeNDAP y recuperar datos estructurados a través de consultas URL.

En la solicitud de datos permite especificar variables, dimensiones y restricciones, lo que le permite trabajar con datos grandes y permitir descargas parciales facilitando un acceso eficiente.

La utilización de PyDAP viene dada por su utilización en la ejecución del algoritmo a la hora de recuperar los datos marítimos para poder modelizar y obtener estos datos que afectan a la ruta en un tiempo concreto.

8.1.3. PyProj

PyProj [119] es una biblioteca Python que se destaca por su capacidad para realizar cálculos geodésicos precisos, incluida la ruta ortodrómica, también conocida como "great circle".

Con esta herramienta es posible calcular y trazar rutas ortodrómicas; además, de calcular distancias, rumbos y puntos intermedios en estas.

Se ha usado esta librería para obtener la ruta ortodrómica y las coordenadas de los puntos intermedios. Se utilizó la versión 3.6.0 .

8.1.4. Scikit-learn

Scikit-learn [120] es una biblioteca de aprendizaje automático (machine learning) de código abierto que se utiliza ampliamente en proyectos Python. Proporciona un conjunto completo de herramientas y algoritmos para tareas comunes de aprendizaje automático, como clasificación, regresión, agrupamiento y selección de características.

Además, se beneficia de la amplia adopción de Python en el campo de la ciencia de datos, lo que permite una integración sencilla con otras bibliotecas populares como NumPy, Pandas y Matplotlib. Con ella se puede evaluar, validar y comparar el desempeño de los modelos generados.

El uso de la librería ha venido dado en la comparativa de los diferentes modelos de predicción de la velocidad SOG llevados a cabo en la Sección 6.3. La versión 1.2.2 fue la utilizada en este proyecto.

8.1.5. XGBoost

XGBoost [121] es una biblioteca de aprendizaje automático muy popular y efectiva a la hora de implementación de los árboles de decisión de refuerzo de gradiente. Su capacidad de implementación optimizada y distribuida a la hora de realizar tareas como clasificación o regresión ha demostrado ser particularmente útil para resolver problemas complejos y grandes conjuntos de datos.

Su utilización viene dada, al igual que Scikit-learn, para predecir la velocidad SOG, en este caso para el modelo de *Extreme Gradient Boosting Regressor* (XGBR). Se utilizó la versión 1.7.5 .

8.1.6. Pymoo

Pymoo [122] es una biblioteca de Python que se enfoca en la optimización multiobjetivo (MOO) y la resolución de problemas de optimización complejos. Esta biblioteca ofrece una amplia gama de algoritmos y herramientas para abordar problemas que involucran múltiples objetivos y múltiples restricciones, lo que permite a los usuarios encontrar soluciones óptimas en el espacio de búsqueda.

Con Pymoo, se pueden implementar y resolver problemas de optimización multiobjetivo utilizando algoritmos genéticos, algoritmos evolutivos o algoritmos basados en enjambre, entre otras técnicas avanzadas. Además, la biblioteca ofrece funcionalidades para el cálculo de frentes de Pareto, el análisis de sensibilidad y la visualización de resultados, lo que facilita la comprensión y la toma de decisiones en problemas de optimización multiobjetivo.

El enfoque modular de Pymoo y su facilidad de uso son dos de sus principales ventajas. Los usuarios pueden crear y personalizar sus propios problemas de optimización y usar los algoritmos que ya están

configurados o crear nuevos algoritmos. La biblioteca es escalable y altamente eficiente, por lo que es adecuada para abordar problemas de optimización de gran tamaño y complejidad.

Su uso ha venido dado a la hora de implementar el algoritmo genético NSGA-II, y la definición de las partes del algoritmo (población,cruce y mutación) definidas en el Capítulo 7, para encapsularlo en la clase Pymoo. La versión utilizada en el proyecto es la 0.6.0 .

8.1.7. Plotly.js

Plotly.js [123] es una biblioteca de visualización de datos basada en JavaScript que cuenta con herramientas poderosas y flexibles para crear gráficos interactivos y visualizaciones personalizadas. Plotly.js se basa en las librerías D3.js y stack.gl, y soporta más de 40 tipos de gráficos, incluyendo gráficos 3D, gráficos estadísticos y mapas SVG.

Plotly.js permite la creación de gráficos de forma sencilla, mediante la utilización de objetos JSON para los datos. Estos gráficos pueden integrarse en las aplicaciones web, y además, permitiendo la interactividad con funcionalidades como zoom, selección u otro tipo de eventos y personalización con cualquier tipo de información. Esto mejora la experiencia del usuario y permite explorar los datos de forma más dinámica. Además, Plotly.js permite exportar los gráficos a diversos formatos, incluso pudiendo descargar una imágen de los mismos.

Se ha hecho uso de esta librería en la parte del cliente para representar las rutas marítimas y datos meteorológicos sobre el mapa. Ayudándonos de los gráficos geoespaciales interactivos y personalizados, como *Scattermapbox* y *Densitymapbox*. Con estas funciones, se han podido representar rutas marítimas y datos meteorológicos en un mapa usando coordenadas geográficas, añadir la información sobre la ruta y la meteorología, además, del uso de colores e intensidades para mostrar la información de forma clara y atractiva. Se ha hecho uso de esta herramienta en la versión v1.58.5 .

8.1.8. Flask

Flask [124] es un microframework desarrollado en Python que ofrece un marco ligero para desarrollar aplicaciones web. Basado en las librerías Werkzeug, Jinja y Click, no impone ninguna restricción extra.

Se ha hecho uso de Flask para la arquitectura del servidor.

8.1.9. Visual Studio Code

Visual Studio Code [125] es un editor de texto de código abierto creado por Microsoft. Se trata de un editor multiplataforma y multilenguaje, que junto sus extensiones, permiten añadir todo tipo de funcionalidad a la hora de editar, ejecutar, control de versiones y depurar código.

Se ha optado por este IDE, debido a las diferentes tecnologías utilizadas y el conocimiento por parte del estudiante de su uso.

8.1.10. Jupyter Notebook

Jupyter Notebooks [126] se trata de un entorno de trabaja de código abierto basando en cuadernos Jupyter en los cuales se puede combinar código ejecutable Python u otros lenguajes (en su kernel), con

texto en formato Markdown o cualquier otro tipo de visualización. Produce un fichero .ipynb en formato JSON.

Dicho entorno se ejecuta a través de un navegador we, lo que podemos considerar, por tanto, un entorno multiplataforma.

Su uso vino dado en las labores previas de extracción de los datos, modelado e implementación del modelo de predicción de velocidad SOG en el Capítulo 6.

8.1.11. Overleaf

Overleaf [127] es un editor web de LaTeX. A través de su aplicación web, se puede directamente editar, compartir y visualizar tus documentos a través del visor PDF integrado.

Su utilización se ha basado en la documentación de la memoria del proyecto.

8.1.12. Astah

Astah Professional [128] es una herramienta de modelado y diseño que permite crear y compartir diagramas en diferentes lenguajes y notaciones, como UML.

Esta herramienta ha sido utilizada para la realización de los distintos diagramas mostrados en las parte de Análisis y Diseño del proyecto. La licencia de uso ha sido proporcionada por la Universidad de Valladolid.

8.1.13. Git

Git [129] se trata de un sistema de control de versiones distribuido y de open-source que permite gestionar los estados de los proyectos de una manera eficiente, y permitiendo el trabajo colaborativo en los proyectos.

8.1.14. Gitlab

Gitlab [130] es una plataforma web open-source que permite gestionar y alojar repositorios y proyectos Git. Además ofrece una plataforma colaborativa de desarrollo de software que facilita el trabajo de DevOps.

En este proyecto se ha utilizado como repositorio de almacenamiento del código y del proyecto, como sistema de control de versiones. Para ello se ha usado el alojamiento que proporciona la Escuela de Ingeniería Informática de la Universidad de Valladolid.

8.1.15. Bootstrap

Bootstrap [131] es un framework basada en HTML, CSS y JS que permite la creación de sitios web responsives y adaptado a cualquier dispositivo.

Cuenta con componentes reutilizables, ofreciendo una forma fácil y sencilla de diseñar la interfaz de sus sitios web.

8.2. Implementación representación visual: Plotly.js

Como se ha comentado en secciones anteriores para llevar a cabo la visualización de las rutas se utiliza la librería *Plotly.js*.

Para ello en primera instancia, cuando cargamos la página de inicio se muestra un mapa basado en *OpenStreetMap* como estilo del mapa. La implementación viene dada por el tipo *Scatter Plot*[1]. A la hora de definir los datos a mostrar en el *Scatter Plot* para mostrar el mapa vacio, los atributos *lat* y *long* serán dos arrays vacios. En la Fig. 8.1 se muestra la función que se encarga de mostrar dicha representación.

```
function mapClear(){
        var map = document.getElementById("map");

        var data = [
          {
            type: "scattermapbox",
            lat: [],
            lon: [],
          },
        ];

        var layout = {
          mapbox: {
            style: "open-street-map",
            bearing: 0,
            center: { lat: 40, lon: -50 },
            pitch: 0,
            zoom: 2.5,
          },
          margin: { r: 0, t: 0, l: 0, b: 0 },
        };

        Plotly.plot(map, data, layout, { showSendToCloud: true });
      }
```

Figura 8.1: Función encargada de la visualización del mapa inicial vacío.

Posteriormente, una vez que se calculan las rutas por parte del algoritmo, para llevar a cabo la representación de estas, se utiliza de nuevo el *Scatter Plot* y esta vez con el modo *markers+lines* mediante el cuál, bajo un mapa de estilo de *OpenStreetMap*, asignando a los atributos *lat* y *long* las coordenadas de las rutas proporcionadas por el algoritmo. Además, con el atributo *text* y *name* mostramos información sobre el timeGrid en ese waypoint y demás datos de la ruta.

Por último, en esta representación de las rutas, se acompaña con los valores del timeGrid para ese

[1] https://plotly.com/javascript/scattermapbox/

espacio de la ruta, es decir, el coste en tiempo de cada celda del grid. Para ello, se utiliza la representación mediante un *Mapbox Density Heatmap* [2]. Mediante la representación de un heatmap nos permite plasmar las diferencias de valor en el grid acorde a las condiciones marítimas; para ello con los atributos *lat* y *long* asignaremos las coordenadas del espacio de ruta y en el atributo z el valor de dicha celda con ese par de coordenadas. En la Fig. 8.2 se muestra la función que se encarga de mostrar dicha representación.

[2] `https://plotly.com/javascript/mapbox-density-heatmaps/`

```
function mapUpdatePopulation(population,gridTime,ds_lat,ds_long){
  var map = document.getElementById("map");


  var data = [
    {
      type: "densitymapbox",
      name: 'timeGrid',
      coloraxis: 'coloraxis',
      radius:2,
      z: gridTime,
      lon: ds_long,
      lat: ds_lat,
    },
  ];


  var nameRoutes=['in time', 'minFuel']

  for(var i=0;i<population.length;i++){
    data.push({
      type: "scattermapbox",
      mode: 'markers+lines',
      name: nameRoutes[i],
      lat: population[i][1],
      lon: population[i][0],
      text: population[i][2],
    });
  }

  var layout = {
    mapbox: {
      style: "open-street-map",
      bearing: 0,
      center: { lat: 40, lon: -50 },
      pitch: 0,
      zoom: 2.5,
    },
    showlegend: false,
    coloraxis: {colorscale: "Viridis"},
    margin: { r: 0, t: 0, l: 0, b: 0 },
  };

  Plotly.plot(map, data, layout, { showSendToCloud: true });
}
```

Figura 8.2: Función encargada de la visualización del mapa, rutas y timeGrid.

Capítulo 9

Pruebas

En este Capítulo se explicarán las diferentes pruebas llevadas a cabo en el código de desarrollo del proyecto.

En este punto es importante destacar que durante el proceso de desarrollo se han ido realizando pruebas de caja negra unitarias, es decir, a cada uno de los componentes del sistema con el objetivo de verificar que cumple con los requisitos funcionales especificados, basado en las entradas y salidas esperadas.

Además de estas, se han llevado a cabo **pruebas de caja negra guiadas por casos de uso**. Este tipo de pruebas se basan en los escenarios de interacción entre los actores y el sistema, tal y como se describen en los casos de uso. Con esto conseguimos verificar el cumplimiento de los requisitos funcionales y asegurar el correcto funcionamiento simulando una interacción real entre los usuarios y el sistema.

A estas pruebas de software, se le añaden diferentes ejemplos para verificar la aplicabilidad y resultados del algoritmo desarrollado.

9.1. Pruebas de caja negra guiadas por caso de uso

La descripción de las pruebas serán descrita en las siguientes tablas. En ellas se refieren al CU, descripción, entrada, salida esperada y obtenida; y si ha habido algún error, las medidas tomadas para corregirlo.

P-01	**Calcular Ruta**
Descripción	El usuario desea visualizar las rutas marítimas óptimas según sus parámetros.
Entrada	▪ El usuario introduce los datos del barco. ▪ El usuario introduce los datos de la ruta. ▪ El usuario selecciona la opción de Ejecutar Algoritmo con los parámetros prefijados.
Salida Esperada	El sistema muestra en un mapa los resultados(rutas) obtenidas, junto a un heatmap de las condiciones meteorológicas. Pasando por encima de la ruta y de las condiciones se puede obtener más información acerca de estas.
Salida Obtenida	OK
Corrección	-

Tabla 9.1: Prueba Calcular Ruta

9.2. Modelo de simulación: Pruebas y Resultados Algoritmo Genético

Se presentan a continuación algunos de los ejemplos de rutas probadas para la comprobación del buen funcionamiento del algoritmo genético junto a la aplicación web, y poder apreciar las rutas plasmadas sobre el mapa generado a partir de *Plotly*.

Se llevaron diversas ejecuciones: sobre diferentes puntos de salida y llegada y con diferentes condiciones marítimas plasmadas en nuestro timeGrid. En ellas buscábamos comprobar la eficacia, tiempo de ejecución, configuración de parámetros del algoritmo y limitaciones del modelo, más allá de una ejecución de caja negra correcta.

Con estas diversas ejecuciones, hemos concluido que los parámetros algorítmicos adecuados para los resultados de las rutas son los siguientes:

Variable	Valor
Upper Bound	8
Lower Bound	8
Population Size	200
Generations	500
Offsprings	20
Tasa Mutación	0.3
Nº waypoints	100

Tabla 9.2: Parámetros del algoritmos seleccionados.

Sin embargo, el usuario puede modificar cualquiera de los anteriores parámetros, lo que permite explorar nuevos espacios de soluciones y nuevas combinaciones que quizá para diferentes tipos de ruta se ajusten mejor en tiempo y resultados.

A continuación se muestran diversos tipos de rutas para que se pueda observar el funcionamiento del algoritmo con esos parámetros y cómo las rutas generadas se plasman en la representación sobre el mapa. Las Fig. 9.1 y 9.2 muestran una ruta desde Nueva York hasta Lisboa. La Fig. 9.3 representa la ruta desde Nueva York hasta Monrovia.

En la Fig. 9.1 la ruta tiene una fecha de salida del 26-05-2022 12:00 hasta el 10-06-2022 12:00. En esta se ha buscado representar una ruta con climatología adversa entre ambos puntos para mostrar cómo el algoritmo evita esa zona.

Para las Fig. 9.2 y 9.3 tienen una fecha de salida del 25-12-2022 12:00 hasta el 15-01-2023 12:00. En estas imágenes se muestran climatologías más tranquilas durante la navegación.

Las tres rutas utilizan el mismo modelo de barco con valor *Ship Width*: 40, *Ship Draft*: 12 y *Ship Length*: 270.

Las rutas representadas en naranja representan aquella que minimiza el tiempo de llegada, y las verdes aquellas que minimiza el consumo de combustible (estos colores pueden cambiar respecto a la versión final del código para mejorar la visualización).

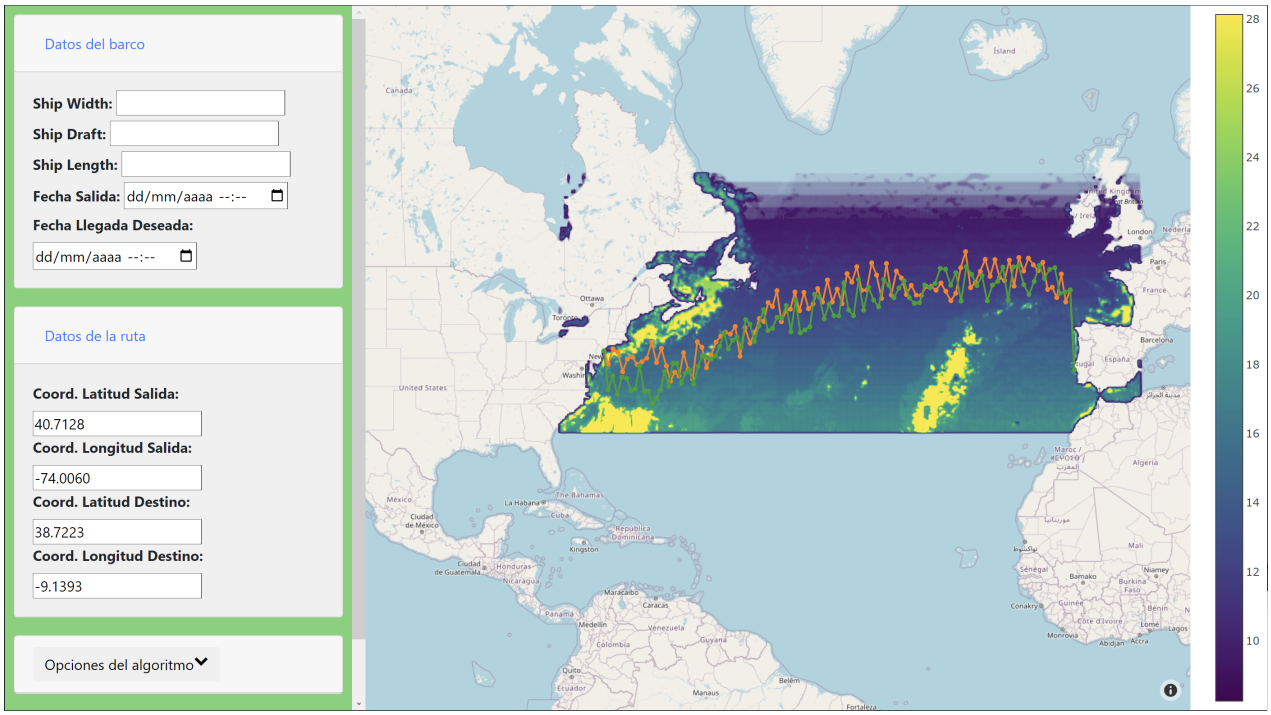

Figura 9.1: Simulación ruta Nueva York-Lisboa 26-05-2022.Condiciones Adversas.

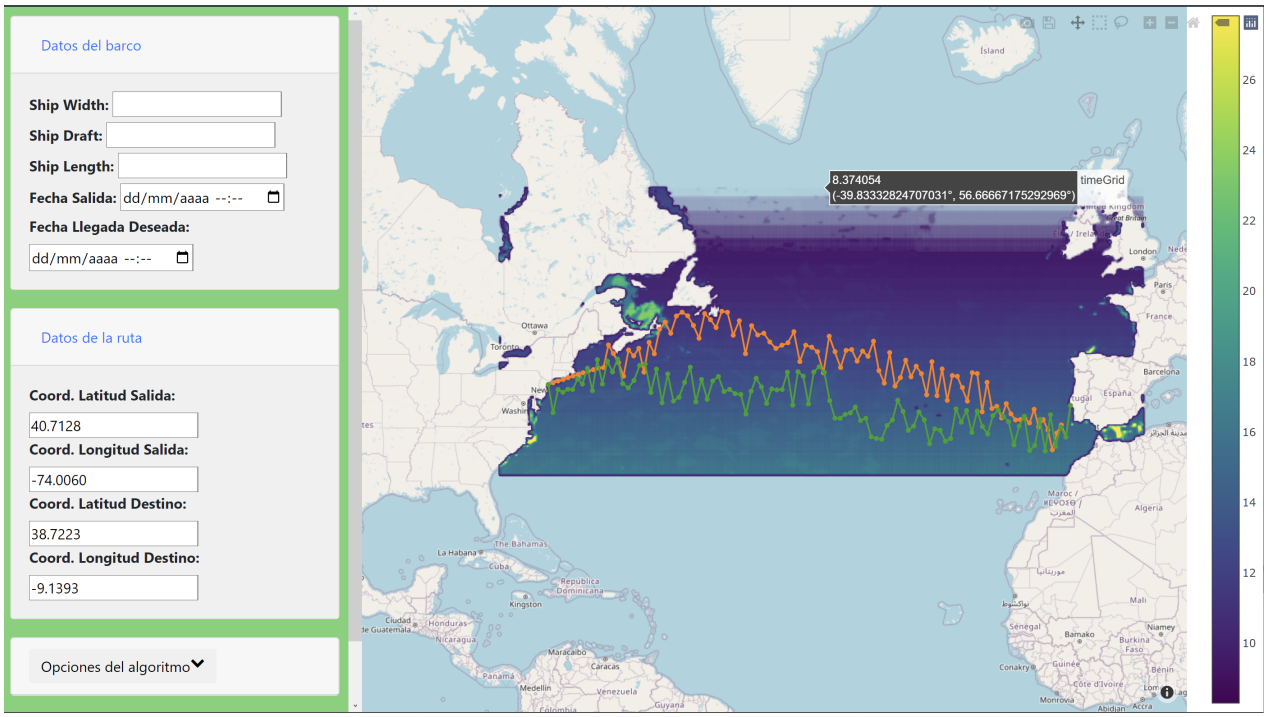

Figura 9.2: Simulación ruta Nueva York-Lisboa 25-12-2022.Condiciones Tranquilas.

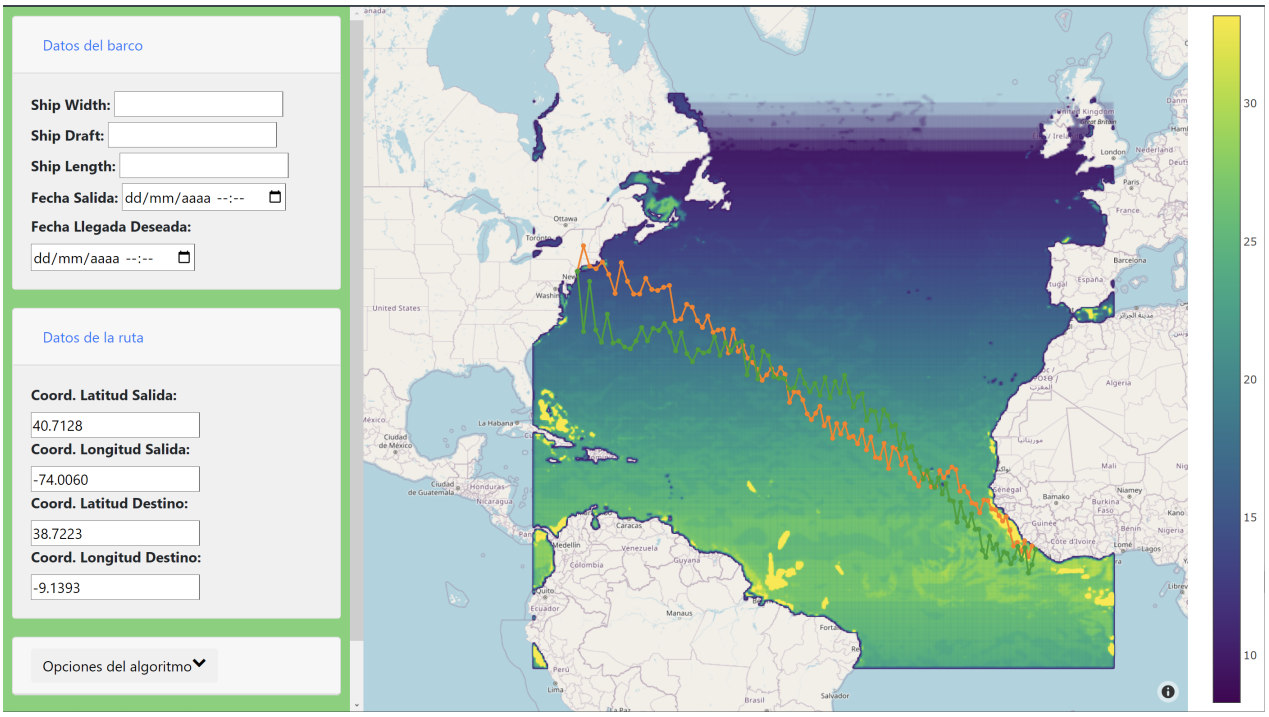

Figura 9.3: Simulación ruta Nueva York-Monrovia 25-12-2022.Condiciones Tranquilas.

Capítulo 10

Conclusiones y Líneas Futuras

10.1. Conclusiones

Este documento ha reflejado el proceso y resultados del Trabajo Fin de Grado del alumno, en el que se ha desarrollado una aplicación web que ha representado de forma visual la investigación, análisis e implementación de un algoritmo genético que resuelve la problemática de la optimización de rutas marítimas teniendo en cuenta la meteorología.

En cuanto a los objetivos establecidos en la Sección 1.3, podemos hacer las siguientes reflexiones:

- Se ha conseguido realizar un análisis e investigación de la problemática asociada a la optimización de rutas marítimas. Este paso ha abarcado tanto la identificación de los problemas asociados (climatológicos, operativos y del buque) como una revisión de las soluciones y trabajos llevados a cabo.
- Extracción, procesado y transformación de los datos marinos para la consecución de la creación de un modelo predictivo de la velocidad sobre el suelo (SOG). Para ello se realizó un estudio de los datos y las características a extraer, la propia extracción y procesado de estos datos para que puedan ser ingeridos por un modelo de aprendizaje automático para la predicción del SOG, y por último, una discusión sobre la evaluación de los diferentes modelos probados.
- Se ha logrado diseñar, desarrollar e implementar un modelo de algoritmo genético para la consecución de la optimización de rutas. Los pasos seguidos han sido la definición de los parámetros de este, estudio y elección del algoritmo genético, implementación mediante la librería *Pymoo* de optimización y unas series de pruebas para ajustar los parámetros para obtener el mejor resultado.
- Se ha logrado realizar un modelo de pruebas de simulación para obtener las conclusiones del desarrollo y el rendimiento del algoritmo a la hora de conseguir el objetivo final.
- Se ha conseguido que el resultado del algoritmo se pueda visualizar a través del diseño de una plataforma web, en la que cualquier usuario pueda interactuar y calcular su ruta óptima. Además, se logró que la representación visual permita la interactividad con la representación de las ruta a la hora de mostrar detalles de estas, se muestre información de las condiciones marítimas, capacidad de ampliar el mapa e incluso descargarse una imagen de la representación de la ruta.

Desde un enfoque técnico se han podido explorar diferentes tecnologías y diversas bibliotecas para el procesamiento y análisis de grandes volúmenes de datos meteorológicos. Asimismo, ha implicado una

revisión exhaustiva de la literatura científica para diseñar un algoritmo adecuado al problema planteado, así como la elaboración de un Estado del Arte sobre el sector marítimo, que era desconocido al inicio del proyecto. Los resultados obtenidos han sido satisfactorios y han demostrado la viabilidad de la propuesta.

Desde una perspectiva personal, este trabajo ha sido una oportunidad de integrar y aplicar los conocimientos adquiridos a lo largo de la carrera de Ingeniería Informática. Dentro del marco en el que se encuadra la temática del proyecto, se desea hacer frente a un problema real de gran envergadura, que está siendo objeto de investigación por parte de múltiples universidades a nivel mundial. Por lo que el reto era enorme, ya que ha exigido un aprendizaje profundo del sector y la búsqueda de soluciones innovadoras, con un alto grado de incertidumbre.

10.2. Líneas de trabajo futuras

Debido a la magnitud del proyecto, esta versión preliminar del estudio de investigación puede ser mejorada y ampliada. Esto supondría obtener soluciones que se puedan acercar más a la realidad y con una mejora a la hora de obtención de estas. Las líneas de trabajo a futuro planteadas son las siguientes:

- Implementar mecanismos que supongan una mejora en la eficiencia computacional a la hora de obtener la ruta óptima. Actualmente, al necesitar lanzar una petición para obtener los datos de Copernicus y modelar el SOG, se requiere de grandes recursos, lo que repercute en la eficiencia computacional y el tiempo de cálculo. Esto ha sido una de las razones que no ha permitido actualizar al tiempo de los productos de datos, los nuevos datos marítimos. Para ello, se plantean dos acciones: una relacionada con la obtención de los datos de Copernicus y otra con el modelado del SOG. La primera consiste en la extracción diaria de los datos marítimos para que puedan almacenarse en una base de datos y su acceso sea más rápido. La segunda consiste en reducir el número de cálculos del SOG, limitándose a los que se encuentren dentro del espacio de la ruta y no al grid completo.

- Mejora a la hora de representación y modelado de la ruta. El resultado actual nos proponen rutas con grandes cambios de rumbo con coordenadas dispersas debido a la modelización de la ruta, el espacio de coordenadas y el planteamiento de los pasos de cruce y mutación del algoritmo genético. Para ello se propone el estudio de la implementación de alguna medida que mediante el ángulo de giro y la distancia sea capaz de encontrar coordenadas más cercanas. Además de esta solución, la implementación de la modelización de la ruta mediante la función *route_through_array* de la librería *scikit-image*[1] podría ayudar a paliar este efecto, ya que tiene en cuenta el costo de todo el grid que atraviesa y generaría rutas más "suavizadas". Otra idea puede surgir entorno al modelo de generación de población, cuyo origen se basa a partir del "great circle", y a partir de este y los UpperBound y LowerBound se generan rutas y coordenadas de estas random dentro de este espacio: quizá se debería introducir dentro de la población rutas más parecidas a la ruta ortodrómica, no con coordenadas tan dispares, pero que exploren este espacio y mediante los procesos de cruce y mutación nos sirvan para refinar las rutas finales.

- Implementación de el algoritmo GA+PSO planteado al inicio del proyecto y cuyo marco teórico está definido en el Capítulo 3 Podría implementarse acompañado del NSGA-II planteado, ya que el patrón de diseño estrategia nos permite un fácil intercambio entre los algoritmos. Esto nos permitiría establecer una comparativa con los resultados del NSGA-II y sacar conclusiones acerca de los resultados y rendimientos de estos.

- A lo largo del TFG se ha explicado cómo la variable potencia juega un papel fundamental a la hora de optimizar la ruta. La capacidad de poder ajustar la potencia del motor permitiría conseguir una

[1] `https://scikit-image.org/docs/stable/api/skimage.graph.html#skimage.graph.route_through_array`

ruta más realista, que se adapte mejor a las condiciones marítimas y por tanto logre un mejor ajuste en las funciones objetivo de minimizar el tiempo y combustible. Para ello se debería ser capaz de modelizar la velocidad a partir de las diferentes variaciones en la potencia del motor cuya fórmula es: $Potenciadelmotor = constante * SOG^3$; y a partir del 70 % de potencia de motor asumida en este trabajo podemos calcular la constante, y por tanto que nuestro algoritmo pueda ir jugando con la potencia de motor como nueva variable.

- A lo largo del Capítulo 3 se han planteado el estudio de diversos modelos teóricos matemáticos para el cálculo de velocidad y consumo de fuel, entre otros. Por lo que podría ser interesante poder estudiar la viabilidad de estos modelos más exactos y comparar la diferencia con los modelos empíricos y de machine learning utilizados.
- Mejora de la interfaz e interactividad en la plataforma web para hacer que el usuario tenga una experiencia más satisfactoria y eficiente con la plataforma.

Bibliografía

[1] M. Simonsen, E. Larsson, W. Mao, and J. Ringsberg, "State-of-the-art within ship weather routing," 06 2015.

[2] J. Seuren, "Multi-objective ship weather routing: An evolutionary approach," Master's thesis, Dec. 2020. [Online]. Available: http://hdl.handle.net/2105/55728

[3] H.-B. Wang, X.-G. Li, P.-F. Li, E. I. Veremey, and M. V. Sotnikova, "Application of real-coded genetic algorithm in ship weather routing," *The Journal of Navigation*, vol. 71, no. 4, p. 989–1010, 2018.

[4] U. Mehboob, J. Qadir, S. Ali, and A. Vasilakos, "Genetic algorithms in wireless networking: Techniques, applications, and issues," *Soft Computing*, vol. 20, 06 2016.

[5] C. A. Correa Flórez, R. Andrés Bolaños, and A. Molina Cabrera, "Algoritmo multiobjetivo nsga-ii aplicado al problema de la mochila." *Scientia Et Technica*, 2008. [Online]. Available: https://www.redalyc.org/articulo.oa?id=84920503037

[6] W. Abd-El-Wahed, A. Mousa, and M. El-Shorbagy, "Integrating particle swarm optimization with genetic algorithms for solving nonlinear optimization problems," *Journal of Computational and Applied Mathematics*, vol. 235, no. 5, pp. 1446–1453, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377042710004796

[7] B. S, "Web applications using python flask - part i." [Online]. Available: https://www.polarsparc.com/xhtml/Python-Flask-1.html (Accessed 10/06/2023).

[8] P. Kennedy, "Deep dive into flask's application and request contexts." [Online]. Available: https://testdriven.io/blog/flask-contexts-advanced/ (Accessed 10/06/2023).

[9] S. A. Amin, "Minimal flask application using mvc design pattern." [Online]. Available: https://medium.com/@arslanaut/minimal-flask-application-using-mvc-design-pattern-842845cef703 (Accessed 10/06/2023).

[10] M. Abebe, Y. Shin, Y. Noh, S. Lee, and I. Lee, "Machine learning approaches for ship speed prediction towards energy efficient shipping," *Applied Sciences*, vol. 10, no. 7, 2020, doi: https://doi.org/10.3390/app10072325.

[11] European Union-Copernicus Marine Service, "Global ocean 1/12° physics analysis and forecast updated daily," 2016. [Online]. Available: https://resources.marine.copernicus.eu/product-detail/GLOBAL_ANALYSISFORECAST_PHY_001_024/INFORMATION

[12] "Sistemas de coordenadas geográficas," April 2021. [Online]. Available: https://www.ibm.com/docs/es/db2woc?topic=SS6NHC%2Fcom.ibm.db2.luw.spatial.topics.doc%2Fdoc%2Fcsb3022a.htm (Accessed 08/06/2023).

[13] W. G. . D. Ltd., "Engine selection for very large container vessels," *WinGD*, September 2016. [Online]. Available: https://www.wingd.com/en/documents/general/papers/engine-selection-for-very-large-container-vessels.pdf/

[14] W. Zhao, Y. Wang, Z. Zhang, and H. Wang, "Multicriteria ship route planning method based on improved particle swarm optimization–genetic algorithm," *Journal of Marine Science and Engineering*, vol. 9, no. 4, p. 357, Mar 2021. [Online]. Available: http://dx.doi.org/10.3390/jmse9040357

[15] U. N. C. on Trade and Development, "Review of maritime transport 2022." [Online]. Available: https://unctad.org/system/files/official-document/rmt2022_en.pdf (Accessed 02/05/2023).

[16] "Reducción de las emisiones de gases de efecto invernadero procedentes de los buques." [Online]. Available: https://www.imo.org/es/MediaCentre/HotTopics/Pages/Reducing-greenhouse-gas-emissions-from-ships.aspx (Accessed 02/05/2023).

[17] "Safety and shipping review 2022: Agcs," May 2022. [Online]. Available: https://www.agcs.allianz.com/news-and-insights/reports/shipping-safety.html (Accessed 02/05/2023).

[18] Z. Perunovic and J. V. –Perunovic, "Innovation in the maritime industry," *POMS Annual Conferences*, no. 22, 2011.

[19] C. Egloff, U. Sanders, J. Riedl, S. Mohottala, and K. Georgaki, "The digital imperative in container shipping," Apr 2023. [Online]. Available: https://www.bcg.com/publications/2018/digital-imperative-container-shipping (Accessed 02/05/2023).

[20] A. Chircop, "The imo initial strategy for the reduction of ghgs from international shipping: A commentary," *The International Journal of Marine and Coastal Law*, vol. 34, no. 3, pp. 482–512, 2019.

[21] J. Panigrahi, C. Padhy, D. Sen, J. Swain, and O. Larsen, "Optimal ship tracking on a navigation route between two ports: A hydrodynamics approach," *Journal of Marine Science and Technology*, vol. 17, pp. 59–67, 02 2011.

[22] D. Technologies, "Pythia: Performance routing for the 21st-century." [Online]. Available: https://www.deepsea.ai/pythia/ (Accessed 04/04/2023).

[23] D. Technologies, "Seanergy achieves fuel savings of up to 12 % through deepsea's industry-first performance routing platform." [Online]. Available: https://www.deepsea.ai/pythia-performance-routing-platform/ (Accessed 04/04/2023).

[24] D. Technologies, "Wallenius wilhelmsen moves to fully ai-based voyage planning with deepsea." [Online]. Available: https://www.deepsea.ai/wallenius-webinar/ (Accessed 04/04/2023).

[25] D. Technologies, "Deepsea technologies: Next-generation weather routing." [Online]. Available: https://www.ship-technology.com/analysis/deepsea-technologies-next-generation-weather-routing/ (Accessed 04/04/2023).

[26] D. Technologies, "Ai in shipping: Challenges and opportunities." [Online]. Available: https://www.deepsea.ai/knowledge_hub/ai-in-shipping-challenges-and-opportunities/?utm_source=linkedin&utm_medium=organic (Accessed 04/04/2023).

[27] DTN, "Spos seakeeping." [Online]. Available: https://www.dtn.com/weather/shipping/spos/ (Accessed 04/04/2023).

[28] DTN, "How torm balances safety and efficiency with spos fleetguard." [Online]. Available: https://www.dtn.com/wp-content/uploads/2021/04/how-spos-fleetguard-helps-balance-safety-and-efficiency-for-torm.pdf (Accessed 04/04/2023).

[29] M. Buitendijk, "Response-based routing for maersk line," Dec. 2013. [Online]. Available: https://swzmaritime.nl/news/2013/12/12/response-based-routing-for-maersk-line/ (Accessed 04/04/2023).

[30] DTN, "How spos seakeeping helps stop containers from getting lost at sea." [Online]. Available: https://www.dtn.com/wp-content/uploads/2021/06/spos-seakeeping-stops-lost-containers.pdf (Accessed 04/04/2023).

[31] S. Ocean, "Sofar ocean - wayfinder." [Online]. Available: https://www.sofarocean.com/products/wayfinder (Accessed 04/04/2023).

[32] G. Egan, "Dynamic route optimization: Finding the shortest path with wayfinder." [Online]. Available: https://www.sofarocean.com/posts/dynamic-route-optimization-finding-the-shortest-path-with-wayfinder (Accessed 04/04/2023).

[33] E. XPRT, "Wayfinder by sofar - voyage optimization." [Online]. Available: https://www.environmental-expert.com/software/wayfinder-by-sofar-voyage-optimization-849586 (Accessed 04/04/2023).

[34] S. M. D. Solution, "Sinay." [Online]. Available: https://sinay.ai/eS/ (Accessed 04/04/2023).

[35] S. M. D. Solution, "How does route planning work?" [Online]. Available: https://sinay.ai/en/sinay-hub/route-planning/ (Accessed 04/04/2023).

[36] K. Schwaber and J. Sutherland, "The 2020 scrum guide." [Online]. Available: https://scrumguides.org/scrum-guide.html (Accessed 04/02/2023).

[37] C. Blog, "What is agile? what is scrum?" [Online]. Available: https://www.cprime.com/resources/what-is-agile-what-is-scrum/ (Accessed 04/02/2023).

[38] A. Andrews, "Scrum of one: How to bring scrum into your one-person operation." [Online]. Available: https://www.kodeco.com/585-scrum-of-one-how-to-bring-scrum-into-your-one-person-operation (Accessed 04/02/2023).

[39] U. de Valladolid, "Guía docente del trabajo de fin de grado (mención en computación)." [Online]. Available: https://albergueweb1.uva.es/guia_docente/uploads/2021/545/46978/1/Documento.pdf (Accessed 08/02/2023).

[40] B. Hughes and M. Cotterell, *Software project management*, 5th ed. McGraw-Hill Higher Education, May 2009.

[41] Asus, "Asus vivobook s 14 oled (k3402,12th gen intel)." [Online]. Available: https://www.asus.com/es/laptops/for-home/vivobook/vivobook-s-14-oled-k3402-12th-gen-intel/ (Accessed 04/04/2023).

[42] Samsung, "Monitor qhd 32" con panel ips y usb tipo-c s32a600uuu." [Online]. Available: https://www.samsung.com/es/monitors/high-resolution/s60ua-32-32-inch-uhd-4k-usb-c-ls32a600uuuxen/?cid=es_pd_ppc_google_Monitors_ongoing_SmartMonitor_pla_Shopping_hot_pfx&awc=21707_1676847351_bb98a9bf2e30ba1ff4d9055ce9653976 (Accessed 04/04/2023).

[43] E. Tu, G. Zhang, L. Rachmawati, E. Rajabally, and G.-B. Huang, "Exploiting ais data for intelligent maritime navigation: A comprehensive survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. PP, 06 2016.

[44] M. Ueno, Y. Yoshimura, Y. Tsukada, and H. Miyazaki, "Circular motion tests and uncertainty analysis for ship maneuverability," *Journal of Marine Science and Technology*, vol. 14, pp. 469–484, 12 2009.

[45] J. Montewka, R. Guinness, L. Kuuliala, F. Goerlandt, P. Kujala, and M. Lensu, *Challenges in modelling characteristics of maritime traffic in winter conditions and new solution proposal*, 10 2017.

[46] R. Guinness, J. Saarimaki, L. Ruotsalainen, H. Kuusniemi, F. Goerlandt, J. Montewka, R. Berglund, and V. Kotovirta, "A method for ice-aware maritime route optimization," in *2014 IEEE/ION POSITION, LOCATION AND NAVIGATION SYMPOSIUM - PLANS 2014*, ser. IEEE-ION Position Location and Navigation Symposium. United States: IEEE, 2014, pp. 1371–1378, iEEE/ION Position, Location and Navigation Symposium (PLANS) ; Conference date: 05-05-2014 Through 08-05-2014.

[47] K. K. Lai and K. Shih, "A study of container berth allocation," *Journal of Advanced Transportation*, vol. 26, pp. 45 – 60, 12 1992.

[48] Y. C. Seong, J. S. Jeong, and G.-K. Park, "The relation with width of fairway and marine traffic flow," *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, vol. 6, no. 3, pp. 317–321, 2012.

[49] N. Bowditch, *The American Practical Navigator*. National Imagery and Mapping Agency, 2002.

[50] DNV GL, "Energy management study 2015." [Online]. Available: https://www.dnv.com/news/dnv-gl-energy-management-study-shows-rise-in-awareness-of-performance-monitoring-26307 (Accessed 03/05/2023).

[51] H. Chen, V. Cardone, and P. Lacey, "Use of operation support information technology to increase ship safety and efficiency," *SNAME Transaction*, vol. 106, pp. 105–127, 01 1998.

[52] Ø. Buhaug, J. Corbett, Ø. Endresen, V. Eyring, J. Faber, S. Hanayama, D. Lee, D. Lee, E. Lindstad, A. Markowska, A. Mjelde, D. Nelissen, J. Nilsen, C. Pålsson, J. Winebrake, W. Wu, and K. Yoshida, "Prevention of air pollution from ships," 04 2009.

[53] T. Notteboom and P. Cariou, "Fuel surcharge practices of container shipping lines: Is it about cost recovery or revenue-making," pp. 24–26, 07 2009.

[54] J. Holtrop and G. Mennen, "An approximate power prediction method," *International Shipbuilding Progress*, vol. 29, pp. 166–170, 07 1982.

[55] E. Işıklı, N. Aydın, L. Bilgili, and A. Toprak, "Estimating fuel consumption in maritime transport," *Journal of Cleaner Production*, vol. 275, p. 124142, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0959652620341871

[56] T. Fujiwara, M. Ueno, and Y. Ikeda, "Cruising performance of a large passenger ship in heavy sea," *Proceedings of the International Offshore and Polar Engineering Conference*, 01 2006.

[57] K. Hans Otto and M. Lützen, "Prediction of resistance and propulsion power of ships," 05 2012, project no. 2010-56, missionsbeslutningsstöttesystem, Work Package 2, report no. 04, Technical University of Denmark and University of Southern Denmark.

[58] Y.-C. Chang, R.-S. Tseng, G.-Y. Chen, P. Chu, and Y.-T. Shen, "Ship routing utilizing strong ocean currents," *Journal of Navigation*, vol. 66, 11 2013.

[59] W. Du, Y. Li, G. Zhang, C. Wang, B. Zhu, and J. Qiao, "Energy saving method for ship weather routing optimization," *Ocean Engineering*, vol. 258, p. 111771, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801822011180

[60] P. Krata and J. Szlapczynska, "Ship weather routing optimization with dynamic constraints based on reliable synchronous roll prediction," *Ocean Engineering*, vol. 150, pp. 124–137, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801817307874

[61] C.-J. Söder, A. Rosén, E. Ovegård, J. Kuttenkeuler, and M. Huss, "Parametric roll mitigation using rudder control," *Journal of Marine Science and Technology*, vol. 18, 09 2013.

[62] I. M. O. , "revised guidance to the master for avoiding dangerous situations in adverse weather and sea conditions," *MSC.1/circ.1228*, 01 2007.

[63] T. P. Zis, H. N. Psaraftis, and L. Ding, "Ship weather routing: A taxonomy and survey," *Ocean Engineering*, vol. 213, p. 107697, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801820306879

[64] M. Sternsson and U. Björkenstam, "Influence of weather routing on encountered wave heights," vol. 49, pp. 85–94, 07 2002.

[65] L. Skoglund, J. Kuttenkeuler, A. Rosén, and E. Ovegård, "A comparative study of deterministic and ensemble weather forecasts for weather routing," *Journal of Marine Science and Technology*, vol. 20, no. 3, pp. 429–441, Nov. 2014. [Online]. Available: https://doi.org/10.1007/s00773-014-0295-9

[66] M. Hoffschildt, J.-R. Bidlot, B. Hansen, and P. Janssen, "Potential benefit of ensemble forecasts for ship routing," Shinfield Park, Reading, p. 25, 08/1999 1999. [Online]. Available: https://www.ecmwf.int/node/9925

[67] P. Chu, S. Miller, and J. Hansen, "Fuel-saving ship route using the navy's ensemble meteorological and oceanic forecasts," *Journal of Defense Modeling and Simulation*, vol. 12, pp. 41–56, 01 2015.

[68] E. Larsson and M. Simonsen, "Direct weather routing," Master's thesis, Chalmers University of Technology, Gothenburg, Sweden, Jan 2014.

[69] R. W. James, "Application of wave forecasts to marine navigation." 1957.

[70] H. Hagiwara and J. A. Spaans, "Practical weather routing of sail-assisted motor vessels," *The Journal of Navigation*, vol. 40, no. 1, p. 96–119, 1987.

[71] Y.-H. Lin, M.-C. Fang, and R. W. Yeung, "The optimization of ship weather-routing algorithm based on the composite influence of multi-dynamic elements," *Applied Ocean Research*, vol. 43, pp. 184–194, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141118713000679

[72] J. Szlapczynska and R. Smierzchalski, "Adopted isochrone method improving ship safety in weather routing with evolutionary approach," *International Journal of Reliability, Quality and Safety Engineering - IJRQSE*, vol. 14, 12 2007.

[73] W. Shao, P. Zhou, and S. K. Thong, "Development of a novel forward dynamic programming method for weather routing," *Journal of Marine Science and Technology*, vol. 17, no. 2, pp. 239–251, Jun 2012. [Online]. Available: https://doi.org/10.1007/s00773-011-0152-z

[74] D. Sen and C. P. Padhy, "An approach for development of a ship routing algorithm for application in the north indian ocean region," *Applied Ocean Research*, vol. 50, pp. 173–191, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141118715000206

[75] G. Mannarini, N. Pinardi, G. Coppini, P. Oddo, and A. Iafrati, "Visir-i: small vessels – least-time nautical routes using wave forecasts," *Geoscientific Model Development*, vol. 9, no. 4, pp. 1597–1625, 2016. [Online]. Available: https://gmd.copernicus.org/articles/9/1597/2016/

[76] T.-N. Chuang, C.-T. Lin, J.-Y. Kung, and M.-D. Lin, "Planning the route of container ships: A fuzzy genetic approach," *Expert Systems with Applications*, vol. 37, no. 4, pp. 2948–2956, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417409008215

[77] L. Wang, Z. Zhang, Q. Zhu, and S. Ma, "Ship route planning based on double-cycling genetic algorithm considering ship maneuverability constraint," *IEEE Access*, vol. 8, pp. 190 746–190 759, 2020.

[78] D. S. Vlachos, "Optimal ship routing based on wind and wave forecasts," *Applied Numerical Analysis & Computational Mathematics*, vol. 1, no. 2, pp. 547–551, 2004. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/anac.200410018

[79] M.-C. Tsou and H.-C. Cheng, "An ant colony algorithm for efficient ship routing," *Polish Maritime Research*, vol. 20, no. 3, pp. 28–38, 2013. [Online]. Available: https://doi.org/10.2478/pomr-2013-0032

[80] G. Zhang, H. Wang, W. Zhao, Z. Guan, and P. Li, "Application of improved multi-objective ant colony optimization algorithm in ship weather routing," *Journal of Ocean University of China*, vol. 20, no. 1, pp. 45–55, Feb 2021. [Online]. Available: https://doi.org/10.1007/s11802-021-4436-6

[81] R. Vettor and C. Guedes Soares, "Detection and analysis of the main routes of voluntary observing ships in the north atlantic," *The Journal of Navigation*, vol. 68, no. 2, p. 397–410, 2015.

[82] R. Vettor and C. Guedes Soares, "Development of a ship weather routing system," *Ocean Engineering*, vol. 123, pp. 1–14, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801816302141

[83] Y. He, D. Zhang, J. fen Zhang, M. Zhang, and T. W. Li, "Ship route planning using historical trajectories derived from ais data," *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, 2019.

[84] Z. Liu, J. Liu, F. Zhou, R. W. Liu, and N. Xiong, "A robust ga/pso-hybrid algorithm in intelligent shipping route planning systems for maritime traffic networks," *Journal of Internet Technology*, vol. 19, no. 6, pp. 1635–1644, 2018. [Online]. Available: https://jit.ndhu.edu.tw/article/view/1785

[85] P. C. Chu, "A genetic algorithm approach for combinatorial optimisation problems," 1997.

[86] A. Eiben and J. Smith, *Introduction To Evolutionary Computing*, 01 2003, vol. 45.

[87] E.-G. Talbi, *Metaheuristics: from design to implementation.* John Wiley & Sons, 2009.

[88] Métodos Matemáticos en Ciencias de la Computación (Universidad del País vasco), "Algoritmos genéticos." [Online]. Available: http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/temageneticos.pdf (Accessed 09/05/2023).

[89] B. Weinberg and E.-G. Talbi, "On search space symmetry in partitioning problems," in *Evolutionary Computation in Combinatorial Optimization*, J. Gottlieb and G. R. Raidl, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 230–240.

[90] A. M. del Río, "Algoritmos genéticos: Cómo funcionan y para qué se utilizan," Jan 2021. [Online]. Available: https://blogs.imf-formacion.com/blog/tecnologia/algoritmos-geneticos-como-funcionan-202010/ (Accessed 09/05/2023).

[91] W. Greene, "Partitioning sets with genetic algorithms." 01 2000, pp. 102–106.

[92] P. C. Chu and J. E. Beasley, "A genetic algorithm for the generalised assignment problem," *Comput. Oper. Res.*, vol. 24, pp. 17–23, 1997.

[93] D. E. Goldberg, J. Richardson *et al.*, "Genetic algorithms with sharing for multimodal function optimization," in *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*, vol. 4149. Hillsdale, NJ: Lawrence Erlbaum, 1987.

[94] S. Tsutsui and A. Ghosh, "A study on the effect of multi-parent recombination in real coded genetic algorithms," 06 1998, pp. 828 – 833.

[95] A. Eiben, P.-E. Raué, and Z. Ruttkay, "Genetic algorithms with multi-parent recombination," 10 1994, pp. 78–87.

[96] Wikipedia, "Selection (genetic algorithm)," Feb 2023. [Online]. Available: https://en.wikipedia.org/wiki/Selection_(genetic_algorithm) (Accessed 09/05/2023).

[97] L. Davis, "Applying adaptive algorithms to epistatic domains," in *Proceedings of the 9th International Joint Conference on Artificial Intelligence - Volume 1*, ser. IJCAI'85. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1985, p. 162–164.

[98] H. Feltl and G. R. Raidl, "An improved hybrid genetic algorithm for the generalized assignment problem," ser. SAC '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 990–995. [Online]. Available: https://doi.org/10.1145/967900.968102

[99] N. Srinivas and K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," *Evol. Comput.*, vol. 2, no. 3, p. 221–248, sep 1994. [Online]. Available: https://doi.org/10.1162/evco.1994.2.3.221

[100] K. Deb and T. Goel, "A hybrid multi-objective evolutionary approach to engineering shape design," in *Evolutionary Multi-Criterion Optimization*, E. Zitzler, L. Thiele, K. Deb, C. A. Coello Coello, and D. Corne, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 385–399.

[101] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learnin," *Cited on*, vol. 33, 2009.

[102] A. Biswal, "Sklearn linear regression (step-by-step explanation): Sklearn tutorial," Apr 2023. [Online]. Available: https://www.simplilearn.com/tutorials/scikit-learn-tutorial/sklearn-linear-regression-with-examples#linear_regression_theory (Accessed 11/05/2023).

[103] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[104] J. Brownlee, "Xgboost for regression," Mar 2021. [Online]. Available: https://machinelearningmastery.com/xgboost-for-regression/ (Accessed 11/05/2023).

[105] J. H. Friedman, "Stochastic gradient boosting," *Computational statistics & data analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[106] S. Mexico, "Escribiendo historias de usuario." [Online]. Available: https://scrum.mx/informate/historias-de-usuario (Accessed 04/02/2023).

[107] Refactoring.Guru, "Patrones de diseño / patrones de comportamiento / strategy." [Online]. Available: https://refactoring.guru/es/design-patterns/strategy

[108] "About : Find out more about copernicus marine service." [Online]. Available: https://marine.copernicus.eu/about (Accessed 15/05/2023).

[109] European Union-Copernicus Marine Service, "Global ocean waves analysis and forecast," 2018. [Online]. Available: https://resources.marine.copernicus.eu/product-detail/GLOBAL_ANALYSISFORECAST_WAV_001_027/INFORMATION

[110] "Guidelines for the onboard operational use of shipborne automatic identification systems (ais)," November 2001. [Online]. Available: https://wwwcdn.imo.org/localresources/en/KnowledgeCentre/IndexofIMOResolutions/AssemblyDocuments/A.917(22).pdf (Accessed 11/05/2023).

[111] A. Harati-Mokhtari, A. Wall, P. Brooks, and J. Wang, "Automatic identification system (ais): Data reliability and human error implications," *The Journal of Navigation*, vol. 60, no. 3, p. 373–389, 2007.

[112] F. A. Sarría, "Coordenadas geográficas," February 2006. [Online]. Available: https://www.um.es/geograf/sigmur/temariohtml/node6_mn.html (Accessed 08/06/2023).

[113] T. Zis, P. Angeloudis, M. G. H. Bell, and H. N. Psaraftis, "Payback period for emissions abatement alternatives: Role of regulation and fuel prices," *Transportation Research Record*, vol. 2549, no. 1, pp. 37–44, 2016. [Online]. Available: https://doi.org/10.3141/2549-05

[114] H. Wang, W. Mao, and L. Eriksson, "Efficiency of a voluntary speed reduction algorithm for a ship's great circle sailing," *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, vol. 14, no. 2, pp. 301–308, 2020. [Online]. Available: https://www.transnav.eu/Article_Efficiency_of_a_Voluntary_Speed_Wang,54,1004.html

[115] W. Niu, H. Sui, Y. Niu, K. Cai, and W. Gao, "Ship pipe routing design using nsga-ii and coevolutionary algorithm," *Mathematical Problems in Engineering*, vol. 2016, p. 7912863, Dec 2016. [Online]. Available: https://doi.org/10.1155/2016/7912863

[116] pymoo, "pymoo: Multi-objective optimization in python. part ii: Find a solution set using multi-objective optimization." [Online]. Available: https://pymoo.org/getting_started/part_2.html

[117] S. Marty and CLStoulouse, "Motu project," https://github.com/clstoulouse/motu#readme.

[118] pydap, "pydapn." [Online]. Available: https://www.pydap.org/en/latest/index.html

[119] pyproj4, "pyproj." [Online]. Available: https://pyproj4.github.io/pyproj/stable/

[120] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[121] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 785–794. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785

[122] J. Blank and K. Deb, "pymoo: Multi-objective optimization in python," *IEEE Access*, vol. 8, pp. 89 497–89 509, 2020.

[123] P. T. Inc. (2015) Collaborative data science. Montreal, QC. [Online]. Available: https://plot.ly

[124] M. Grinberg, *Flask web development: developing web applications with python.* .ºReilly Media, Inc.", 2018.

[125] Microsoft, "Visual studio code." [Online]. Available: https://code.visualstudio.com/

[126] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, and C. Willing, "Jupyter notebooks – a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Schmidt, Eds. IOS Press, 2016, pp. 87 – 90.

[127] Overleaf, "Overleaf." [Online]. Available: https://www.overleaf.com/

[128] Astah, "Software design tool – astah professional –." [Online]. Available: https://astah.net/products/astah-professional/

[129] Git, "Git." [Online]. Available: https://git-scm.com/

[130] Gitlab, "Gitlab." [Online]. Available: https://about.gitlab.com/

[131] Bootstrap, "Bootstrap." [Online]. Available: https://getbootstrap.com/

Apéndice A

Manual de instalación

En el siguiente manual de instalación se explicará paso a paso cómo instalar la aplicación web creada para la optimización de rutas marítimas.

A.1. Prerrequisitos

Se requiere de un entorno Python3. El usado en el proyecto ha sido la versión 3.10.11. Puede ser descargado en `https://www.python.org/downloads/`.

El sistema operativo utilizado en el proyecto, y los comando que se describen de instalación, están orientados a Windows. Sin embargo, el proyecto puede ser ejecutado en cualquier sistema operativo.

Además de contar con el lenguaje Python, para el uso y funcionamiento del proyecto se debe contar con una cuenta de Copernicus (CMEMS). Esta es necesaria para poder descargar los datos marítimos que afectan a la ruta y poder calcular la ruta óptima. La obtención de la cuenta es gratuita y se puede obtener en el siguiente enlace: `https://data.marine.copernicus.eu/register`.

El usuario además deberá contar con conexión a Internet, ya que será necesaria para la llamada para poder descargar los datos.

A.2. Acceso y uso de la aplicación

Para poder hacer uso de la aplicación desarrollada y su puesta en funcionamiento se deberían ejecutar los siguientes pasos:

1. Clonar el repositorio a su máquina personal con el siguiente comando:

   ```
   git clone https://gitlab.inf.uva.es/migchav/tfg-miguel-chaveinte.git
   ```

2. Crear un fichero *.env* que contenga la cuenta de Copernicus con la siguiente estructura:

   ```
   USER_CMEMS='usuario'
   PASS_CMEMS='contraseña'
   ```

3. Crear un entorno virtual para el proyecto:

   ```
   pip install virtualenv
   ```

   ```
   python -m venv .venv
   ```

   para activar dicho entorno:

   ```
   .venv\Scripts\Activate.ps1
   ```

   Si se utiliza Visual Studio Code como editor de código se recomienda leer el tutorial de Flask para Visual Studio Code en `https://code.visualstudio.com/docs/python/tutorial-flask` para activar el entorno virtual.

4. Importar todas las dependencias del proyecto:

   ```
   pip install -r requirements.txt
   ```

5. Ejecutar la aplicación:

   ```
   python -m flask run
   ```

6. Acceder a la aplicación a través del navegador en la URL:

   ```
   http://127.0.0.1:5000
   ```

   ó

   ```
   http://localhost:5000/
   ```

Apéndice B

Manual de usuario

Una vez que se accede a la página web nos encontraremos la pantalla inicial, en la que al lado izquierdo encontraremos un formulario con los datos del barco, la ruta y las opciones del algoritmo; junto con el botón de ejecutar algoritmo. En la parte derecha podemos ya ver un mapa de OpenStreetMap, como podemos ver en la imágen Fig. B.1.

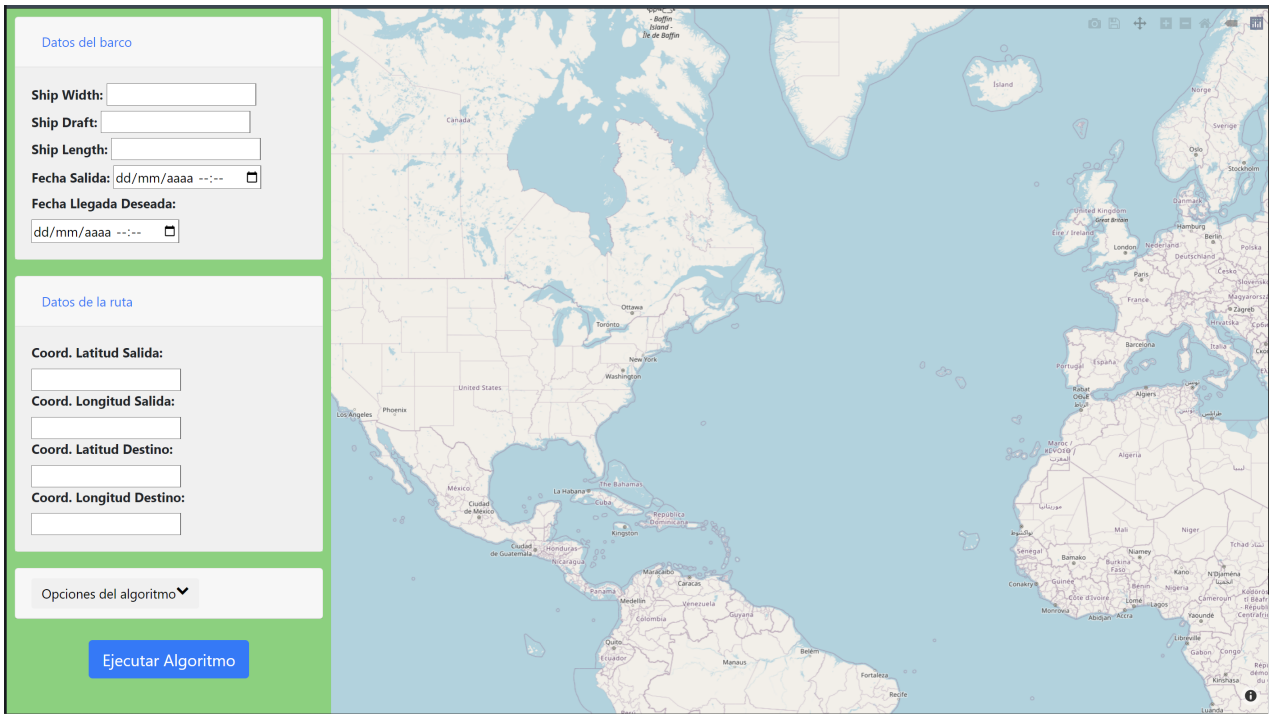

Figura B.1: Pantalla de inicio.

Como comentábamos anteriormente, el usuario debe introducir los datos del barco del barco y de la ruta para poder ejecutar el algoritmo:

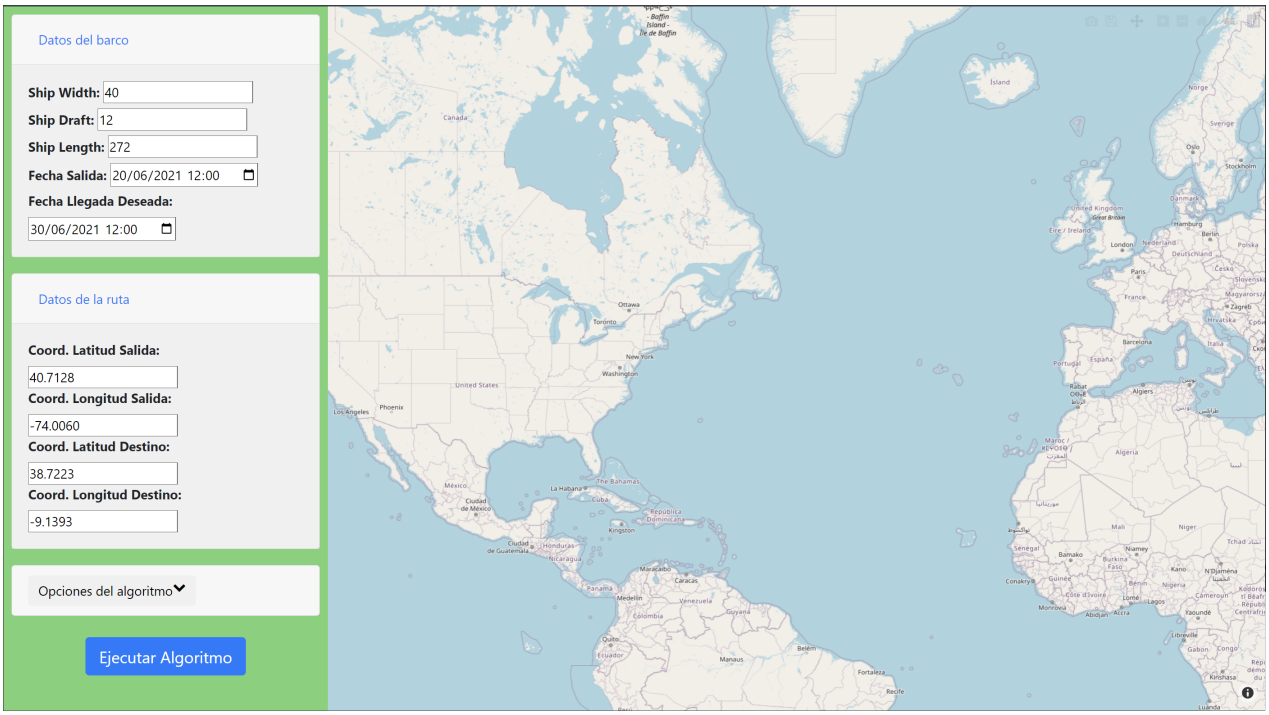

Figura B.2: Pantalla de inicio con datos proporcionados.

Además existe la posibilidad de modificar los parámetros por defecto (prefijados por ser los que mejor resultado nos han proporcionado) con los que funciona el algoritmo :

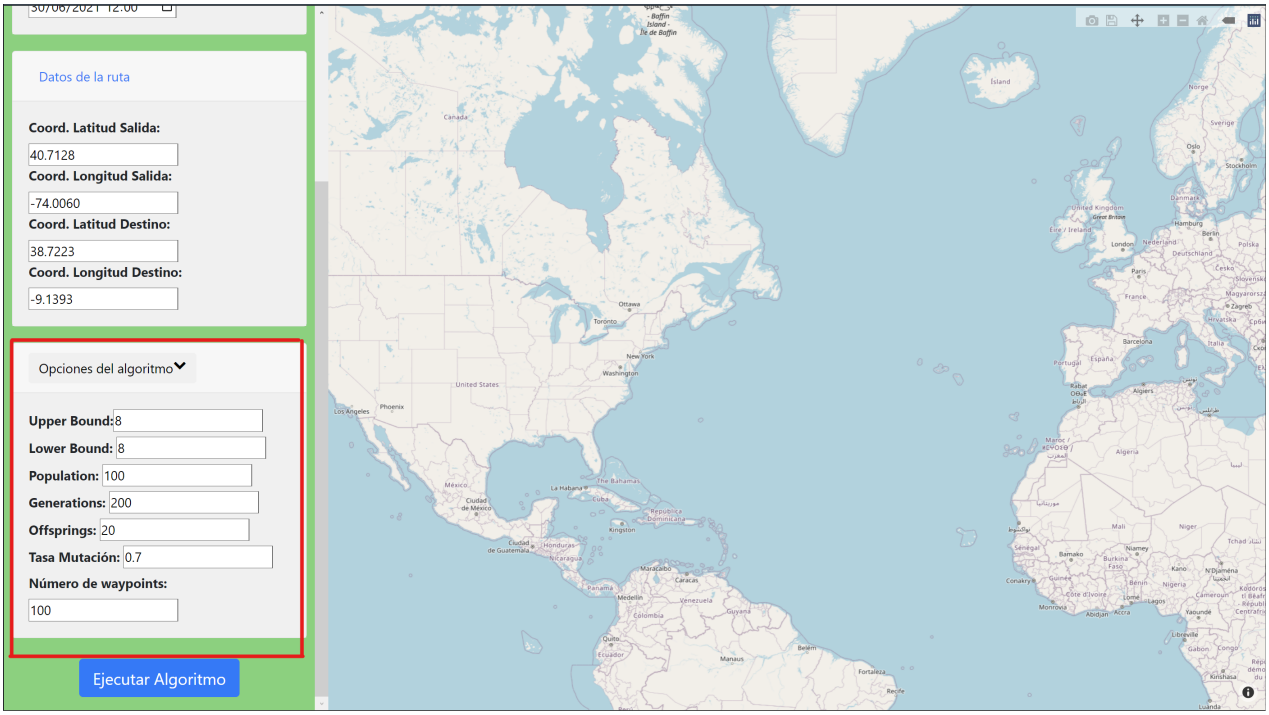

Figura B.3: Opciones de modificación de los parámetros del algoritmo.

Una vez que se da al botón de Ejecutar Algoritmo, este lanza una llamada para obtener las rutas óptimas según los dos objetivos planteados. El resultado se muestra en la Fig. B.4 y B.5. En la segunda, además de mostrar las rutas, se muestra el detalle del waypoint de la ruta cuando se pasa el ratón por

encima. Se muestra las coordenadas entre paréntesis: (longitud,latitud); y en la siguiente línea el valor del timeGrid en ese punto; y en la parte derecha de la etiqueta del pop-up el nombre referencia de la ruta y a qué objetivo minimiza.

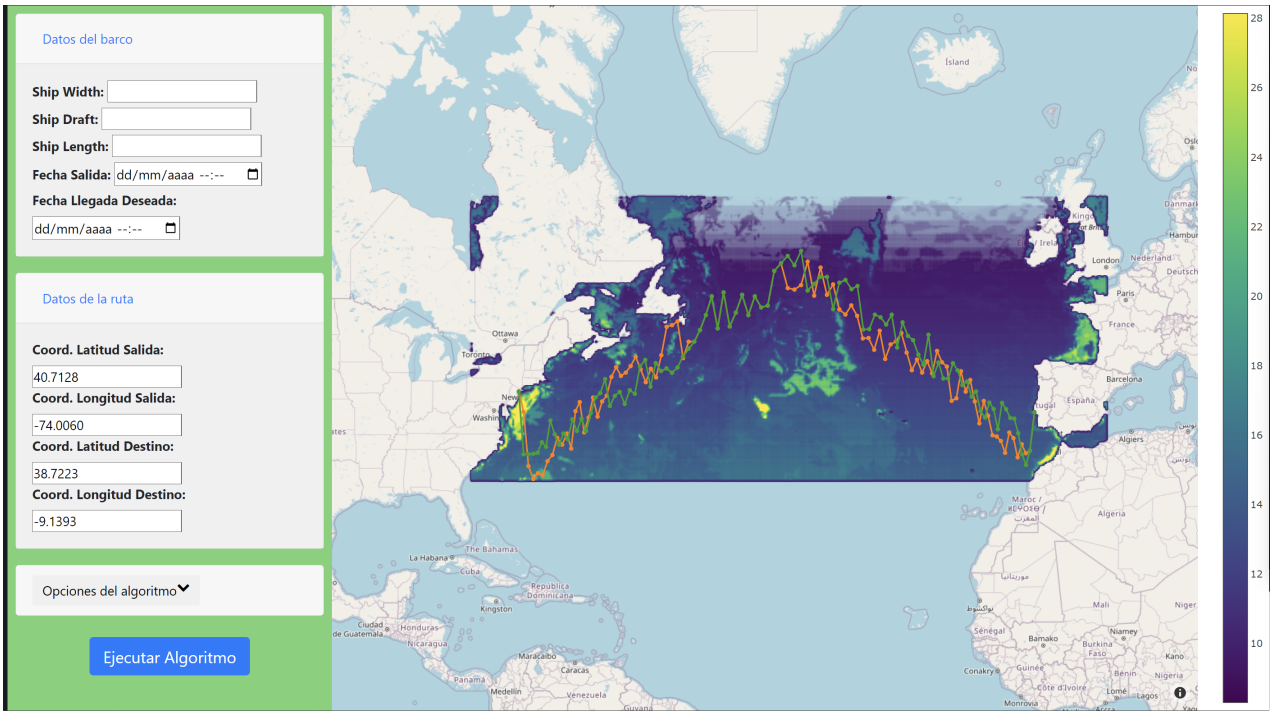

Figura B.4: Resultado de la visualización de las rutas óptimas.

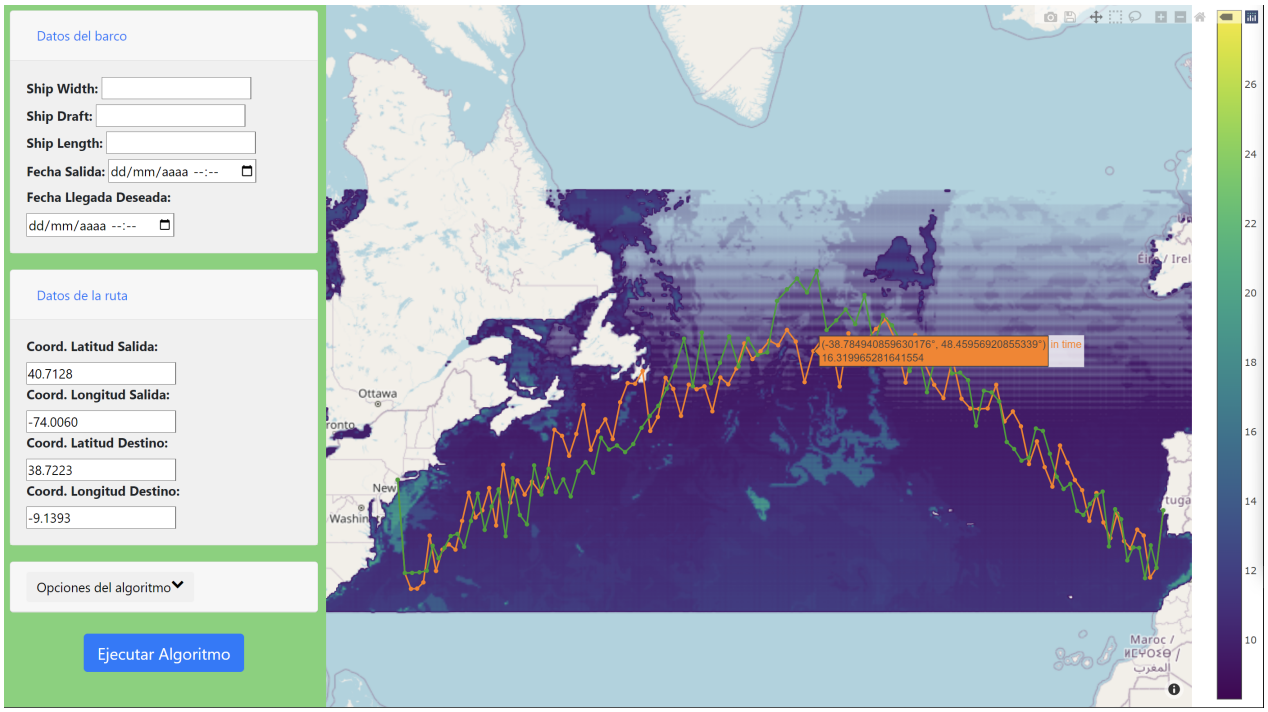

Figura B.5: Detalle mousehover sobre las rutas.

De la misma forma, para obtener la descripción del espacio de ruta y el valor del timeGrid:

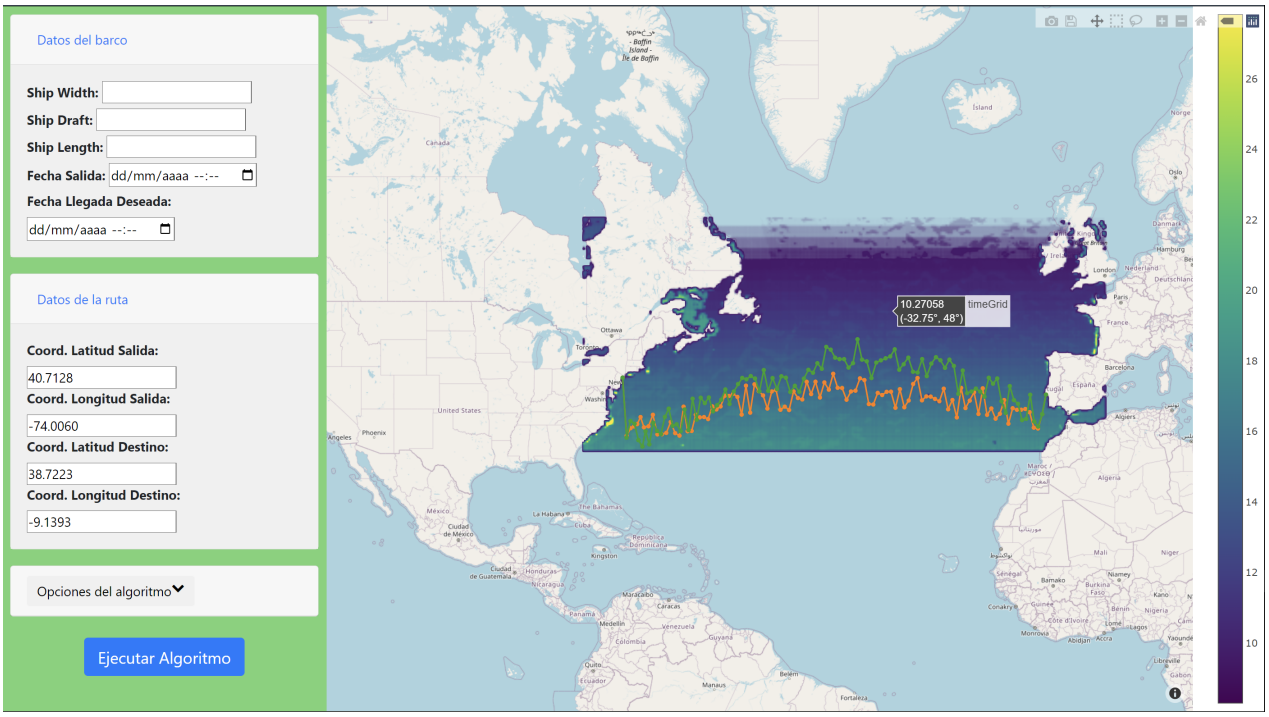

Figura B.6: Detalle mousehover sobre el grid del espacio de ruta.